

From: CN=Henry Mirolyuz/O=ChubbMail
Sent: Tuesday, June 7, 2011 4:12 PM
To: CN=Tony Zhang/O=ChubbMail@ChubbMail
Cc: CN=Marina Maevska/O=ChubbMail@ChubbMail; CN=Patrick F Sullivan/O=ChubbMail@ChubbMail; CN=Peter Tribulski/O=ChubbMail@ChubbMail
Subject: Re: Blaze business rules engine implementation
Attach: brmsstrategyassessmentforchubbcoefinal.pdf

Hi Tony,

I am sending you a document prepared by FICO for Chubb which provide an overview of and recommendations for Business Rules management strategy. This document is quite large but I would recommend reading sections regarding Rules harvesting and lifecycle management.

If you have any questions, please let me know.

Thanks,

Henry

Henry Mirolyuz | It Technical Analyst | Chubb Business Rules CoE
 82 Hopmeadow Rd | Simsbury, CT 06070 | W: (860) 408-2428 | C: (860) 294-6866
 | * hmirolyuz@chubb.com

From: Henry Mirolyuz/ChubbMail
 To: Tony Zhang/ChubbMail@ChubbMail
 Cc: Peter Tribulski/ChubbMail@ChubbMail, Marina Maevska/ChubbMail@ChubbMail, Patrick F Sullivan/ChubbMail@ChubbMail
 Date: 05/16/2011 10:44 AM
 Subject: Re: Blaze business rules engine implementation

Hi Tony,

My apologies for the late response. I think it is definitely worth having a session to discuss best practices regarding BR development. My calendar is up to date so feel free to schedule a meeting at your convenience. Please also invite Patrick Sullivan to this session.

Thanks,

Henry

Henry Mirolyuz | It Technical Analyst | Chubb Business Rules CoE
 82 Hopmeadow Rd | Simsbury, CT 06070 | W: (860) 408-2428 | C: (860) 294-6866
 | * hmirolyuz@chubb.com



Confidential

FED007128_0001

From: Tony Zhang/ChubbMail
To: Jennifer L Reed/ChubbMail@ChubbMail
Cc: Peter Tribulski/ChubbMail@ChubbMail, Marina Maevska/ChubbMail@ChubbMail,
Henry Mirolyuz/ChubbMail@ChubbMail
Date: 05/05/2011 01:05 PM
Subject: Blaze business rules engine implementation

Hi Jennifer,

We have started a Work Manager project to manage our business processes and harvest business rules from existing legacy code, Peter Tribulski referred your name as business rules domain lead.

We had a session with Henry regarding Blaze rules engine capability in the past, and Henry also helped us with Blaze software installation for our rules engine Proof Of Concept. As a business rules engine, Blaze seems to be a good fit for our requirements.

Can we have a session to discuss BRE (Business Rules Engine) best practices and guidelines you may already have? and also we can talk about some of our requirements and get some design ideas from your team.

Please let me know what time works for you, and don't hesitate to contact me if you have any questions.

Thanks!

Tony Zhang | IT Architect
Chubb Insurance Company of Canada | (: 416-359-3222 ext. 4006 | *: tonyzhang@chubb.com
Chubb Canada IT – Business Excellence Through Technology

THIS E-MAIL (WHICH INCLUDES ANY ATTACHMENTS) IS INTENDED TO BE READ ONLY BY THE PERSON(S) TO WHOM IT IS ADDRESSED. THIS E-MAIL MAY CONTAIN CONFIDENTIAL, PROPRIETARY INFORMATION. IF YOU HAVE RECEIVED THIS E-MAIL IN ERROR, DO NOT PRINT IT, FORWARD IT OR DISSEMINATE IT OR ITS CONTENTS. IN SUCH EVENT, PLEASE NOTIFY THE SENDER BY RETURN E-MAIL (OR BY PHONE AT THE NUMBER SHOWN ABOVE) AND DELETE THE E-MAIL IMMEDIATELY THEREAFTER. THANK YOU FOR YOUR CO-OPERATION.

whitepaper

FICO

Business Rule Management Systems Strategy Assessment and Recommendations

Prepared for:



**CHUBB GROUP
OF INSURANCE COMPANIES**

March 29, 2011

DELIVERABLE DOCUMENTATION

Prepared for, and delivered to Chubb & Son on March 31, 2011

Pursuant to Statement of Work Change Orders Nos. 04 and 05
of the Appendix 2, Statement of Work dated November 11, 2009,
by and between Chubb & Son and Fair Isaac Corporation,
a.k.a. 2010 Retainer Agreement

Prepared by:

FICO™

901 Marquette Avenue
Suite 3200,
Minneapolis, Minnesota 55402
USA

Tel: +1 (612) 758 5200

Fax: +1 (612) 758 5201

E-mail: info@FICO.com
www.FICO.com

© 2011 Fair Isaac Corporation. All rights reserved. No part of this paper may be reproduced, transmitted, or disclosed in any form or by any means, electronic or mechanical, including photocopying and recording, for any purpose without the express written permission of Fair Isaac.

www.fico.com

Make every decision count™

Confidential

FED007129_0001



table of contents

1	Executive Summary	7
1.1	Overview	7
1.1.1	FICO Company Profile	8
1.1.2	Overview of Blaze Advisor	9
1.1.3	FICO Team	12
1.2	Overview of Content	14
1.2.1	BRMS Value Statement	14
1.2.2	BRMS in Personal Lines Insurance Industry	14
1.2.3	FICO's Methodology for Decision Harvesting	15
1.2.4	Rule Taxonomy Recommendations for CPI	16
1.2.5	Personal Lines Use Cases	16
1.2.6	Business Rule Lifecycle Management	17
2	Business Rule Management System (BRMS) Value Statement	19
2.1	Overview	19
2.2	BRMS Value to the Enterprise	19
2.2.1	BRMS Value to the Application SDLC	20
2.2.2	BRMS Value to the Business	21
2.2.3	BRMS Value to IT	22
2.2.4	BRMS Value Assessment	22
2.3	Benefits vs. Added Complexities for the Business Community	25
2.3.1	BRMS Elevator Speech for Business	25
2.3.2	Business Challenges/Drivers for BRMS Capabilities	25
2.3.3	Benefits for the Business Communities	26
2.3.4	Added Complexities for the Business Communities	27
2.4	Benefits vs. Added Complexities for the IT Community	27
2.4.1	BRMS Elevator Speech for IT	28
2.4.2	Technology Challenges/Drivers for BRMS Technologies	28
2.4.3	Benefits for the IT Communities	28
2.4.4	Added Complexities for the IT Communities	30
2.5	ROI and Value throughout the BRMS Application Lifecycle	30
2.5.1	BRMS Industry Assertions about Reducing the Cost of Development	31
2.5.2	Reinvest the Value from Faster Development into Better Development	31
2.6	FICO Recommended Future Strategies for Decision Management at Chubb	32
2.6.1	Get Rid of the Silos	32
2.6.2	Bring Equations to the Planning Table	33
2.6.3	All-points fraud protection	33
3	BRMS in Personal Line Insurance Industry Overview	34
3.1	Overview	34
3.2	Current Trends for BRMS in the Insurance Industry	34
3.2.1	Case Studies of BRMS at Some Major Personal Lines Insurance Carriers	35
3.3	Historical Successes—Things That Make BRMS Successful	37
3.3.1	Select the Right Application	38
3.3.2	Follow a Methodology	39
3.3.3	Document, Document, Document	39
3.3.4	Manage Traceability	40
3.3.5	Manage Business Rule Quality	40
3.3.6	Choose the Right Metaphor	42
3.3.7	Verify the Business Rules	44
3.3.8	Validate Business Rules	45
3.3.9	Simulate the Business Impact	45
3.3.10	Structure for Reuse and Governance	46
3.3.11	Operationalize Analytics and Improve Decisions	47



3.4	Historical Challenges—Things That Add Risk to BRMS Projects	48
3.4.1	Overview	48
3.4.2	Challenging Domains	48
3.4.3	Challenging Business Requirements	49
3.5	BRMS Application Suitability—Guidelines for Selecting App for BRMS	50
3.5.1	Overview	50
3.5.2	Assessment of Volatility and Complexity of Decisioning	50
3.5.3	Indicators and Contraindications for BRMS	52
3.5.4	Sources of Knowledge	52
3.5.5	Types of Decisioning	54
3.5.6	Types of Functional Requirements	56
4	Overview of FICO's Decision Harvesting Methodology	59
4.1	Overview	59
4.2	FICO's FIRUP Delivery Methodology	60
4.2.1	Iterative Project Management	60
4.2.2	FIRUP Disciplines	62
4.2.3	Key Disciplines for Decision Requirements Analysis	63
4.3	FICO's Decision Harvesting Methodology	70
4.3.1	Decision Requirements Harvesting Techniques	70
4.3.2	Effective Requirement Reviews	76
4.3.3	Qualities of "Good" Requirements	77
4.3.4	Guidelines for Writing Good Requirements	77
4.3.5	Outline Requirements	78
4.3.6	Detail Requirement	79
4.3.7	Identifying Architecturally Significant Requirements	80
4.3.8	FIRUP Guidelines for Conducting a Requirements Workshop	80
4.3.9	Traceability of Decisioning Requirements	83
4.3.10	Review and Approve Requirements	86
4.4	Decision Requirements Analysis	87
4.4.1	Overview of DRA	87
4.4.2	Inputs to DRA	88
4.4.3	Outputs Produced from DRA	88
4.4.4	Preconditions and Triggers	88
4.4.5	Resources and Responsibilities	89
4.4.6	Preparation	89
4.4.7	Process Steps	89
4.5	Decision Requirements Analysis Workshop	90
4.5.1	Resources	90
4.5.2	Scheduling	91
4.5.3	Technique	91
4.5.4	Process	91
4.5.5	Outlining Decision Requirements without DRAW	101
4.6	Decision Requirements Diagram (DRD)	102
4.6.1	Components of a DRD	103
4.6.2	Uses of the DRD	104
4.7	Decision Definition Document (DDD)	106
4.7.1	Purpose and Scope	106
4.7.2	Description of Major Sections	106
4.8	Supporting Requirements Document	110
4.8.1	Supporting Requirements Categories	111
4.9	Decision Harvesting (a.k.a., Rule Harvesting)	115
4.9.1	Resources and Responsibilities	116
4.9.2	Inputs	116
4.9.3	Outputs	117
4.9.4	Preparation	117
4.9.5	Process Steps	117
4.9.6	Capture and Document the Rules	117
4.9.7	Capture and Document the Business Terms associated with each Rule	118



FICO Recommendations Whitepaper for Chubb Business Rule COE

4.9.8	Capture and Document Decision Dependencies	120
4.9.9	Capture and Document Abstractions	121
4.9.10	Analyze Rules for Consistency, Completeness and Redundancies	122
4.9.11	Review and Obtain Approval/Sign-off on Rules	122
4.10	Decision Harvesting Workbooks	123
4.10.1	Purpose and Scope	123
4.10.2	Business Term Catalog	123
4.10.3	Decision Dependencies Worksheet	124
4.10.4	Generic Rule Worksheet	124
4.10.5	Structured Rule Worksheet	124
4.10.6	Computation Worksheet	125
4.10.7	Abstraction Worksheet	125
4.11	Ancillary Rule Harvesting Activities and Artifacts	125
4.11.1	Unit Test Requirements Definition	126
4.11.2	User Acceptance Test (UAT) Requirements Definition	126
4.11.3	Rule Maintenance and Extension Requirements	127
4.11.4	RMA Requirements	127
4.11.5	Reporting, Logging, Audit, and Traceability Requirements	128
5	Rule Mining Recommendations for CPI Print and Underwriting	129
5.1.1	Manual Rule-Mining Recommendations	129
5.2	Overview of Rule Mining	130
5.2.1	Business Rule Mining vs. Association Rule Mining	130
5.3	Business Rule Mining Process Steps	130
5.3.1	Define Project Goals and Desired Outputs	131
5.3.2	Select Rule Mining Technology	131
5.3.3	Define Project Approach and Time/Resources Allocation	133
5.3.4	Define Rule Mining Workflow	134
5.3.5	Determine Business Processes Mapping and Alignment	134
5.3.6	Perform Application Analysis	135
5.3.7	Define Business Term Catalog	136
5.3.8	Define Rule Taxonomy	136
5.3.9	Mine Business Rule Requirements (a.k.a., Candidate Rules)	137
5.3.10	Verify and Validate Candidate Rules	140
5.3.11	Transform Business Rule Requirements	141
5.3.12	Reports Production	143
5.3.13	Integration with a Target Environment	143
5.3.14	Proactively Manage Rules	144
5.4	Rule Mining Recommendations for Chubb	145
5.4.1	Some Rules of Thumb about Rule Mining	145
5.4.2	Manual Rule-Mining Recommendations	146
5.4.3	Subject-matter Experts (SME)	147
5.4.4	Subject-matter Expertise Is Needed for Rule Mining	148
6	Rule Taxonomy Recommendations for Chubb Personal Lines (CPI)	150
6.1	Overview	150
6.2	There are Dozens of Rule Taxonomies, and They Aren't All Wrong	150
6.2.1	Enterprise/Organizational Taxonomy vs. BRMS Application/Project Taxonomy	150
6.2.2	Classification vs. Organizational/Structural	153
6.2.3	Taxonomy for the Enterprise or Application/Project	154
6.3	FICO's Approach to Rule Taxonomy	155
6.3.1	Guidelines for Selecting a Rule Taxonomy	155
6.3.2	FICO's Rule Taxonomy	155
6.4	FICO Recommendations for CPI Rule Taxonomy	156
6.4.1	FICO Recommendations for Underwriting Taxonomy	156
6.4.2	FICO Recommendations for Print Taxonomy	156



7	Personal Lines Insurance Rule Use Cases	159
7.1	Overview	159
7.2	BRMS and Underwriting for Personal Lines Insurance	159
7.2.1	New Focus for IT/IS: Revenue Generation	160
7.2.2	Making Revenue-Generating Decisions at Point of Sale	161
7.2.3	Getting a Better Handle on the Channel	163
7.2.4	Multiplying Profits and Markets through More Accurate, Innovative Pricing	164
7.2.5	Facilitating Profitable Renewals	167
7.2.6	Expanding the Market by Tightening the Market	168
7.3	Decision Management Is the Next Step	169
8	Business Rule Lifecycle Management	171
8.1	Overview	171
8.2	Blaze Advisor Product Features and Proper BRMS Structure and Design	172
8.2.1	Overview	172
8.2.2	Blaze BRMS Component Structure	173
8.2.3	Blaze Repository Physical Structure	174
8.2.4	Blaze Repository Logical Structure	175
8.2.5	User Authentication and Authorization (A&A)	177
8.2.6	Lifecycle Management for Decision Service	180
8.2.7	Lifecycle Management for Repository Entities	181
8.2.8	Filters	181
8.2.9	Management Properties (Repository Entity Metadata)	183
8.2.10	Difference Query	184
8.2.11	Graphical Difference Tool	186
8.3	FICO's Best Practices for Rule Versioning	187
8.3.1	Overview	187
8.3.2	Version Management System Considerations	188
8.3.3	Versioning Rules, Rulesets, and Decision Metaphors	188
8.3.4	Effective Dating	191
8.4	FICO Best Practices for Business Rule Lifecycle Management (BRLM)	192
8.4.1	Overview	192
8.4.2	Blaze Advisor Product Features for BRLM	192
8.4.3	BRLM Migration Patterns	192
8.4.4	BRLM Use Case	195
8.4.5	Real-world Considerations—Balancing Governance and Business Agility	198
8.4.6	Impact of SDLC on BRLM	199
8.4.7	Summary	200
8.5	FICO Lifecycle Management Service Framework	200
8.5.1	Overview	200
8.5.2	FICO Lifecycle Management Console Walkthrough	207
8.5.3	FICO Lifecycle Management Lifecycle Database	223
8.5.4	Lifecycle Management Rule Service	233
8.5.5	Best Practices and Considerations	245
8.6	The "Maker – Checker" Approach	247
8.6.1	Overview	247
8.6.2	"Maker—Checker" Approach	248
8.6.3	Sample "Maker—Checker" Use Case	248
8.6.4	Best Practices and Design Considerations	248
8.6.5	Issues and Design Considerations with Sample Use Case	249
8.7	FICO Recommendations for CPI Rule Management	250
8.7.1	Overview	250
8.7.2	Review of Onsite Findings	250
8.7.3	FICO Recommendations for Chubb Enterprise	255
8.7.4	FICO Recommendations for Underwriting Rule Management	257
8.7.5	FICO Recommendations for Print Rule Management	257
8.7.6	FICO Recommendations for Chubb Specialty Lines	257



FICO Recommendations Whitepaper for Chubb Business Rule COE

Attachments and Additional References	258
COBIT Management Strategies Whitepaper	258
Notices/ Disclaimers	259
Confidentiality	259
Agreement	259
Validity	259
Trademarks	259



1 Executive Summary

This whitepaper is provided to Chubb & Son as a Deliverable pursuant to Statement of Work Change Orders Numbers 04 and 05 to Appendix 2 Statement of Work (SOW) for Services dated November 11, 2009, a.k.a., "The 2010 Retainer Agreement."

This whitepaper satisfies the following Deliverables as specified in the description of Services for the above:

- i. BRMS Value Statement—an executive-level overview of the value that BRMS brings to an enterprise.
- ii. Personal Lines Insurance Industry Overview—an executive-level overview of BRMS usage in the Personal Lines Insurance industry.
- iii. Rule Taxonomy for Chubb Personal Lines—a distillation of the Business Rules Taxonomy from Chubb's Corporate Business Rule Management Center of Excellence (COE) for Chubb Personal Lines Insurance (CPI). Note: the content of this section changed to FICO Recommendations for a Rule Taxonomy.
- iv. Rule Harvesting Methodology—an overview of FICO's FIRUP Methodology for Traceability of Decisioning Requirements as it applies to the analysis and definition of decisioning requirements (a.k.a., Rule Harvesting).
- v. Underwriting Rules Use Cases—a catalogue of executive-level use cases for hypothetical scenarios for the application of BRMS for Chubb Personal Insurance's Underwriting.
- vi. Rule Maintenance & Versioning Strategy—FICO's recommended strategy for Rule Maintenance and Versioning for CPI.
- vii. Extensions of Rule Maintenance & Versioning Strategy for Chubb Specialty Insurance (CSI).

1.1 Overview

FICO is pleased to provide Chubb with this whitepaper to assist the Chubb & Son Business Rule Management Center of Excellence in their ongoing efforts to define, cultivate, and disseminate a strategy for Business Rule Management Systems (BRMS) throughout Chubb.

The major sections of this whitepaper are organized to align with the deliverables called for in the SOW, with some rearrangement on order to accommodate prerequisite subject matter. The major sections include:

1. Business Rule Management System (BRMS) Value Statement
2. BRMS in Personal Line Insurance Industry Overview
3. Overview of FICO's Decision Harvesting Methodology
4. Rule Mining Recommendations for CPI Print and Underwriting
5. Rule Taxonomy Recommendations for Chubb Personal Lines (CPI)
6. Personal Lines Insurance Rule Use Cases
7. Business Rule Lifecycle Management

Where the executive overviews of most whitepapers would provide a high-level overview of the findings, this whitepaper is more of an instructional paper. Therefore, this Executive Overview section will be mostly to summarize the content of the whitepaper. The overview subsection of the executive overview will provide some prerequisite background for the material herein. This will allow an executive to



understand what has been delivered and allows the meat of the material to assume the reader has some background in BRMS.

The subsections of this overview to the Executive Overview are as follows:

1. FICO Company Profile—we will start this overview with a brief high-level profile of FICO for the benefit of a reader who may be unfamiliar with who FICO is and what products and services we provide.
2. Overview of Blaze Advisor—the next section will provide a brief high-level overview of Blaze Advisor for the benefit of a person unfamiliar with Blaze Advisor the necessary foundation for understanding subsequent subsections. This subsection is written for an executive with a somewhat technical background.
3. FICO Team—profile and credentials of the FICO Principals who made this whitepaper possible

1.1.1 FICO Company Profile

FICO (NYSE:FICO) is the leader in Decision Management, transforming business by making every decision count. We use predictive analytics to help businesses automate, improve and connect decisions across organizational silos and customer lifecycles.

Clients in 80 countries work with FICO to increase customer loyalty and profitability, cut fraud losses, manage credit risk, meet regulatory and competitive demands, and rapidly build market share. Most leading banks and credit card issuers rely on FICO solutions, as do insurers, retailers, health care organizations and other companies. Through the www.myfico.com Web site, consumers use the company's FICO® scores, the standard measure of credit risk, to manage their financial health.

We have pioneered the development and application of critical technologies behind advanced Decision Management. These include predictive analytics, business rules management and optimization. We use these technologies to help businesses improve the precision, consistency and agility of their complex, high-volume decisions.

Our products and services include:

- *Decision Management applications*, which use Decision Management software to automate, improve and connect decisions based on our proprietary analytic models and user-defined strategies. Each of these solutions is built with the analytics, data, decision strategies and process flow for a particular vertical industry and decision area, such as fraud control or credit account management.
- *Analytics*, including predictive models, optimization and portfolio analytics. Our analytics - whether delivered as standalone custom engagements or embedded in our applications - bring new precision to decisions.
- *Decision Management tools*, which give businesses the ability to create their own decision management applications and models. Our tools include business rules management, model development and decision optimization.
- *Scoring solutions*, which deliver powerful predictions of consumer behavior to help businesses make faster, more profitable decisions. Among our analytic products is the FICO® score, which drives billions of credit and marketing decisions a year, and is recognized as the standard measure of US consumer credit risk.
- *Professional services*. We use our deep decision area experience and technical expertise to help individual clients solve unique business challenges, and get more value from our tools and solutions.

FICO works with more than 5,000 businesses worldwide and our technology serves thousands more through our partnerships. We serve global markets through offices in 12 countries.



FICO clients include:

- Nine of the top 10 companies in the Fortune 500
- Two-thirds of the top 100 banks in the world
- 90 of the 100 largest financial institutions in the U.S., and all the 100 largest U.S. credit card issuers
- More than 300 personal and commercial lines insurers in North America and Europe, including the top 10 US personal lines insurers
- 100+ retailers and general merchandisers, including half of the top 50 U.S. retailers
- 150+ healthcare and life sciences companies, including 8 of the world's top 10 pharmaceuticals companies

1.1.2 Overview of Blaze Advisor

The FICO® Blaze Advisor® business rules management system is ranked #1 in sales and sophistication by industry analysts and other third parties. It was recently named Best Software Product by Yphise using ISO 9001:2000-certified methodology, and Best Business Rules Management System by InfoWorld.

Blaze Advisor provides two distinct approaches to creating, visualizing, and editing rules: one aimed at technical IT staff and one at business analysts. Rules defined by either approach can be viewed and maintained in the other. Rules are grouped together into rule sets that determine the outcome of a particular decision, with the possibility for a single rule to belong to multiple rule sets. By providing effective visualization tools and the ability to construct complex rule sets from simple rules, Blaze Advisor aims to increase the complexity of decision making that an organization might choose to manage automatically before having to revert to human decision makers.

Rules and rule sets are all stored in Blaze Advisor's repository, where full version management and change control can be applied. A high-performance runtime engine is provided for .NET and Java environments. Rule sets can be deployed directly to this runtime engine, and the same rules can be deployed unchanged to both Java and .NET. Blaze Advisor also supports COBOL, but in this event the rule sets are compiled into COBOL code. The higher-level application that needs to use the decision service calls the rules engine using the native language construct, or alternatively as a Web service. Monitoring capabilities provide an insight into the execution of rule sets and permit a degree of tuning of the way the rules are executed. Audit capabilities are also provided.

A recently available SmartForms option permits rules to be deployed to client-side Ajax rich Web applications, where a typical use is the application of complex form-filling rules without the need for constant time-consuming interactions with the server.

Figure 1 shows the high-level architecture of Blaze Advisor.

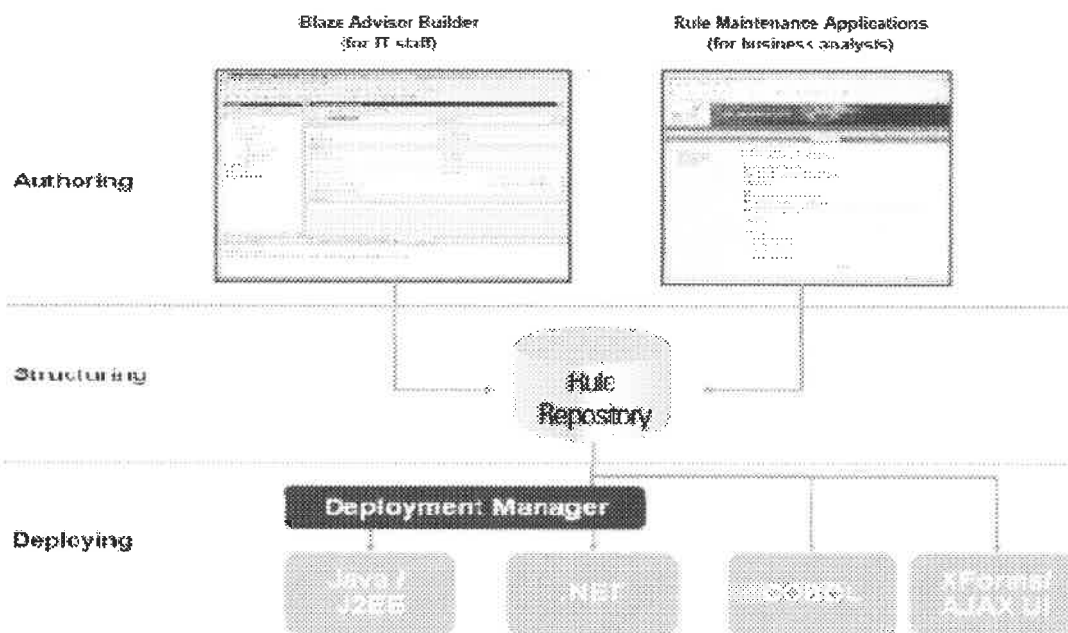


Figure 1—Blaze Advisor Component Architecture

Blaze Advisor is often deployed as a tactical solution to a specific requirement for complex decision making, particularly in circumstances where the rules are subject to frequent change. However, it is more than capable of delivering strategic centralized management of decisions both for server-based applications and front-end client applications, and is a particularly good fit with the strategic adoption of Service Oriented Architecture (SOA).

The operation of Blaze Advisor is best described broken down into the three layers of authoring, structuring, and deploying as shown in Figure 1.

1. Authoring Decisions (Rules)
2. Structuring Decisions
3. Deploying Decision Services

1.1.2.1 Authoring Decisions (Rules)

In most complex decision scenarios there will be activities that can only be carried out by IT staff, such as the integration of the rules environment with information sources. However, the logical definition of how a decision is to be made has to involve a business analyst or business executive, and Blaze Advisor provides capabilities to give these business-focused staff as much autonomy as possible in the creation and maintenance of rules.

The initial definition of a decision is best accomplished by graphical modeling, and FICO provides a simple flow-charting capability where the logical steps in the decision can be entered and prioritized. Figure 2 shows a simple example of a decision model for a loan application.

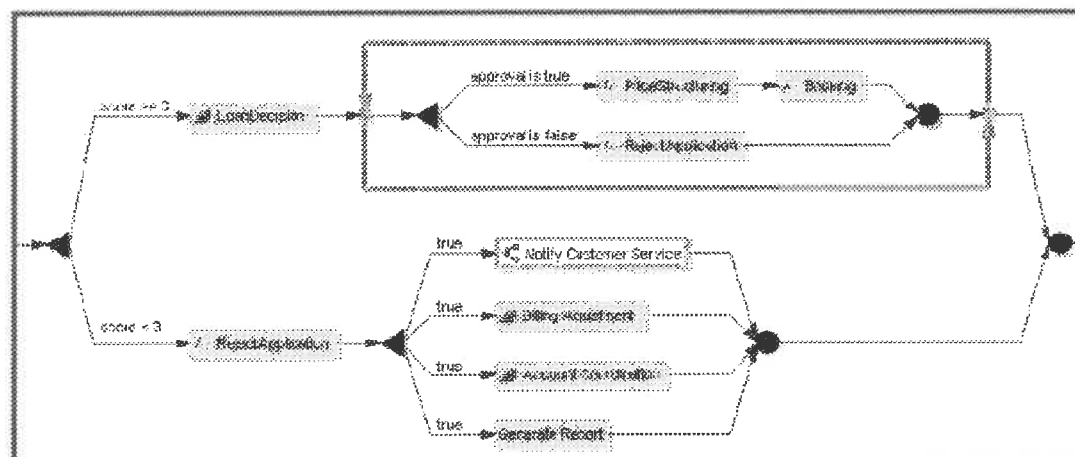


Figure 2—High Level Decision Process Model (Ruleflow)

Each of the steps can then be expanded into rules, writing them in a pseudo-English syntax that is easily understood without technical expertise. The rules related to a particular decision are grouped into a rule set. These steps are achieved using the Blaze Advisor Builder, intended for use by IT staff. Interestingly, although the initial model implies a natural sequence of decision steps, Blaze Advisor will use inference technology within each rule set to calculate the optimum sequence of evaluation to minimize the response time and system overhead. If a particular decision requires additional information in order to be evaluated, Blaze Advisor can use backward chaining to proactively gather the needed information from source systems, but will only commit to this additional work if the decision cannot be fulfilled using the information available.

Blaze Advisor provides multiple views of rules and rule sets aimed at presenting the information in a manner that is easy to comprehend by business users. This clarity and ease of comprehension is an important differentiator for Blaze Advisor. These views include decision tables, graphical decision trees, scorecards, and templates. Templates can be created to show the relevant attributes and allow the business user to directly enter the values and results required. This could include time-related rules, for example, the creation of different price calculation rules for a limited-period special offer. Although it is most likely that templates will be used to allow business users to maintain existing rules, they can also be used to permit rule creation by non-IT staff.

In addition to Blaze Advisor, FICO also offers preconfigured applications based on this technology for certain common industry problems such as fraud detection, payments optimization, and debt management. These are delivered with the rules and rule sets already populated (but user customizable).

1.1.2.2 Structuring Decisions

Rules and rule sets are stored in the repository with full version control. Before deployment of a new or altered rule set, organizations will wish to establish procedures to control the quality of the resulting logic. Blaze Advisor provides two quality assessment tools: rule verification and rule validation. Rule verification establishes that rules are unique to avoid the problems associated with redundancy, and checks to see if all combinations of conditions have been tested. It identifies errors that result in “always true” or “always false” results, as well as infinite loops and other logical errors. Rule verification is a



static test performed on the information in the repository. Rule validation provides regression testing and dynamic validation by executing the rules and rule sets against test data to ensure the anticipated results are delivered. Both of these types of tests can be initiated by business users as well as IT staff.

Blaze Advisor provides full change-management controls including authorization checking, version promotion, and reversion. It supports an intermediate staging system between the development and production environments, with different levels of approval for moving versions between stages. Once ready for deployment rules can be implemented without suspending the rules service. In-flight transactions will continue to use the old version until they complete, while new transactions will dynamically pick up the new version. Additionally, rules can be time-limited and expire automatically at the end of the valid period.

1.1.2.3 Deploying Decision Services

Blaze Advisor provides several execution algorithms that can be selected according to the dynamics of a particular decision. The default uses the Rete algorithm, a public algorithm that has been recognized for nearly three decades as providing efficient management of very large rule sets by organizing the rules into networks instead of sequential execution. Derived from this original algorithm, Blaze Advisor also offers Rete III, which is proprietary to FICO. Because Rete is memory-constrained when used with large datasets there are limits to its general applicability, with the execution time increasing disproportionately with the size of the data set. The Rete III algorithm has been designed to provide linear execution-time increases with both increasing data set and increasing rule set sizes. It therefore represents the most effective option where both large rule sets and large data sets are required.

Where the rule sets are less complex, the sequential execution option can be more efficient. This executes rules in their priority sequence. Finally, a cost option provides compiled sequential execution. This provides just-in-time compilation of the rules, which execute in priority sequence. The compiled sequential option provides the fastest performance for small rule sets against very large data sets.

The runtime environment executes on either .NET or Java Enterprise Edition (JEE) application servers, and uses the underlying infrastructure to provide distribution and fault-tolerance features.

1.1.3 FICO Team

This whitepaper was prepared for Chubb by the following FICO subject-matter experts:

Donald P. Tallo

Principal Consultant, Solution Consulting

Decision Solutions Consulting Practice

Solution and Technology Consulting Group, Americas (STCGA)

FICO™

DonTallo@FICO.com

603-261-9367

Don Tallo is a Principal Consultant with the Decision Solutions Consulting Practice in FICO's Solution and Technology Consulting Group (STCGA). An Electrical Engineer by training, Don has 25 years of experience in consulting and professional services since beginning his career with Burns & McDonnell Engineers and Consultants in 1986. Don became a member of the FICO™ technical team when he joined Neuron Data (a.k.a. Blaze Software) as a Preferred Service Provider in 1993. Since then, he has focused his career exclusively on the development of Business Rule Management System (BRMS) solutions using Blaze Advisor® (and its predecessor technologies). Don has amassed more than 20,000 client-billable hours delivering BRMS solutions to over 300 clients and customers of Neuron Data, Blaze Software, Brokat Technologies, HNC Software, and FICO™.



FICO Recommendations Whitepaper for Chubb Business Rule COE

Raymond J. Bielec*Senior Consultant, Solution Consulting**Decision Solutions Consulting Practice**Solution and Technology Consulting Group, Americas (STCGA)***FICO™**RayBielec@FICO.com

315-420-5645

Ray Bielec is a Senior Consultant with the Decision Solutions Consulting Practice in FICO's Solution and Technology Consulting Group (STCGA). He has a Bachelors degree in Philosophy (logic), Master of Arts - Logic & Scientific Method and (ABD All but Dissertation for PhD). He also completed an eight month training program at Stanford in Knowledge Engineering. He possesses more than 31 years of diversified information system experience, including over 20 years of departmental and project management experience and 21 years in the Insurance Industry where he managed all aspects of information system operations, development and maintenance including managing offshore resources. His background includes fourteen years consulting as a Senior Project Manager, Management Consultant and as a Senior Business Rules Methodology Expert with direct line responsibility for Methodology development, curriculum development and instruction, Business Rule Harvesting and documentation mentoring Business Rule Analysis, defining requirements for Business Rule Repositories etc.

Maarten van Lier*Principal Consultant, Technical Architecture**Decision Solutions Consulting Practice**Solution and Technology Consulting Group, Americas (STCGA)***FICO™**MaartenVanLier@FICO.com

675-576-7376

Maarten van Lier is a Principal Consultant with the Decision Solutions Consulting Practice in FICO's Solution and Technology Consulting Group (STCGA). With a background in almost exclusively Rules based technology, Maarten has designed, architected and implemented countless implementations of Business Rules Management Systems for a large variety of clients. Maarten joined Blaze Software in January of 2000 where he has focused on Blaze Advisor Implementations. Besides having a client facing rule Maarten has also focused on capturing best practices and teaching these internally to many of the Blaze professionals currently active in the field of Blaze Advisor.

Michael L. Sawyer*Associate Partner, Client services**Insurance and Healthcare Industries**Sales and Marketing***FICO™**MikeSawyer@FICO.com

617-589-4951

Mike Sawyer is a Client Partner in FICO's Insurance and Healthcare Practice focused on consulting with insurers and healthcare payors to evaluate and deploy FICO's suite of Decision Management technologies, including predictive analytics, business rule management and decision optimization,

March 31, 2011

FICO Confidential Page 13 of 259

Confidential

FED007129_0013



across the insurance value chain to drive improved business performance. Prior to joining FICO, Mike served in a management role at BearingPoint Management & Technology Consultants delivering a wide range of consulting engagements to leading insurers covering business and technology strategy, operational process improvement, project management, knowledge management and organizational transformation. Throughout his consulting career, Mike's clients have included leading domestic and international property & casualty, life, health and reinsurance carriers. Early in his career Mike also served in various operational positions in underwriting, claims and product development at Sun Life Financial. Mike holds a B.A. in economics from the University of Michigan and is based in Boston.

1.2 Overview of Content

1.2.1 BRMS Value Statement

The BRMS Value Statement is an executive-level overview of the value that BRMS brings to an enterprise.

For companies to be speed-competitive, the logic that drives business decisions can't be buried inside individual software systems. It must be visible, easily modified by nonprogrammers, and usable by any application and channel.

Business rules management (BRM) segregates the logic governing operational decisions from individual applications, where it had been locked within programming code — similar to the way database management separates data. Business rules management delivers substantial savings in application time to market and total cost of ownership.

Business rules can be anything your organization uses to make an operational decision. They might include enterprise, divisional, corporate and line of business policies, as well as calculations and formulas, risk thresholds and regulatory authorizations.

Rules are often expressed—in conversation, written text and software—as “if, then” statements: “If the applicant does not have a sufficient credit history, then pull a report from a debit bureau.”

Rules management solutions such as Blaze Advisor use a similar intuitive and readable syntax to specify and change rules in decisioning applications. Business users can therefore change how an application operates — immediately and without requiring IT help — just by modifying or replacing rules.

This section provides an executive-level overview of the value that a Business Rule Management System (RMS) brings to an enterprise.

This BRMS Value Statement is broken down into the following sections

1. BRMS Value to the Enterprise
2. Benefits vs. Added Complexities for the Business Community
3. Benefits vs. Added Complexities for the IT Community
4. ROI and Value throughout the BRMS Application Lifecycle
5. FICO Recommended Future Strategies for Decision Management at Chubb

1.2.2 BRMS in Personal Lines Insurance Industry

This section provides an executive-level overview of BRMS usage in the Personal Lines Insurance Industry. The subsections include current trends for BRMS in the Insurance Industry, historical successes, historical challenges, and guidelines for conducting suitability analysis on projects.

The subsections of this chapter cover the following subject-matter as it pertains to the application of BRMS in Personal Lines Insurance:



1. Current Trends for BRMS in the Insurance Industry
 - a. Case Studies of BRMS at Some Major Personal Lines Insurance Carriers
 - b. BRMS and Underwriting for Personal Lines Insurance
2. Historical Successes—Things That Make BRMS Successful
 - a. Select the Right Application
 - b. Follow a Methodology
 - c. Document, Document, Document
 - d. Manage Traceability
 - e. Manage Business Rule Quality
 - f. Choose the Right Metaphor
 - g. Verify the Business Rules
 - h. Validate Business Rules
 - i. Simulate the Business Impact
 - j. Structure for Reuse and Governance
 - k. Operationalize Analytics and Improve Decisions
3. Historical Challenges—Things That Add Risk to BRMS Projects
 - a. Challenging Domains
 - b. Challenging Business Requirements
4. BRMS Application Suitability—Guidelines for Selecting App for BRMS
 - a. Assessment of Volatility and Complexity of Decisioning
 - b. Indicators and Contraindications for BRMS
 - c. Sources of Knowledge
 - d. Types of Decisioning
 - e. Types of Functional Requirements

1.2.3 FICO's Methodology for Decision Harvesting

The purpose of Decision Harvesting is to identify and capture the decisioning requirements for the given project or iteration. These requirements form the basis of communication and agreement between the stakeholders and the project team regarding the project's or iteration's deliverables.

Requirements gathering is a fairly broad topic and the detailed processes will vary significantly based upon the type of solution being considered. The FIRUP Methodology is therefore intentionally written to provide a very high-level set of expectations around requirements gathering.

Early in the project, as part of planning sessions, it may be necessary to define high-level requirements in such a manner as to identify the functional increments and architectural components relevant to the overall project. These functional increments would be assigned to specific iterations. Within each iteration, the assigned functional increment would go through further requirements definition to expand on the details.

FICO's decision harvesting methodology is designed to

- Document specifications for decisioning requirements in a manner appropriate structured for implementation within Blaze Advisor



- Provide traceability to ensure that all decision-based requirements are met
- Maximize the maintainability and extensibility of the business rules.

FICO's methodology is highly iterative, JAD-like, and involves a frequent ongoing interactions with business subject-matter experts throughout the project's inception and elaboration phases (sometimes, when requirements are complex or extensive, well into the construction phase).

This section provides an overview of FICO's FIRUP Methodology as it applies to the analysis and definition of decisioning requirements. This overview includes the following subject-matter:

1. FICO's FIRUP Delivery Methodology
2. FICO's Decision Harvesting Methodology
3. Decision Requirements Analysis
4. Decision Requirements Analysis Workshop
5. Decision Requirements Diagram (DRD)
6. Decision Definition Document (DDD)
7. Supporting Requirements Document
8. Decision Harvesting (a.k.a., Rule Harvesting)
9. Decision Harvesting Workbooks
10. Ancillary Rule Harvesting Activities and Artifacts

1.2.4 Rule Taxonomy Recommendations for CPI

Taxonomy is, by definition, the practice and science of classification. The purpose of taxonomy for business rule is simply to document the fact that This_Rule is a This_Kind of rule and That_Rule is a That_Kind of rule during the rule discovery and analysis process.

The term "Rule Taxonomy" is very loosely applied in the BRMS industry. Any sort of schema for classifying or organizing business rules as they are discovered will usually end up being called a Business Rule Taxonomy by the methodologist who devised the schema. However, taxonomy is not supposed to be used as an approach to organizing rules, just classifying them.

There are dozens of different views on rule taxonomy and the subject is debated with much fervor amongst the BRMS methodologist. The purpose of this whitepaper is not to take sides or recommend one over another. The goal of this paper is simply to provide an overview of the approach and to provide Chubb with guidance in selecting one.

The subject-matter covered by the subsections under this section include:

1. There are Dozens of Rule Taxonomies, and They Aren't All Wrong
2. FICO's Approach to Rule Taxonomy
3. FICO Recommendations for CPI Rule Taxonomy
 - a. FICO Recommendations for Underwriting Taxonomy
 - b. FICO Recommendations for Print Taxonomy

1.2.5 Personal Lines Use Cases

This section provides several executive-level use cases for hypothetical scenarios for the application of BRMS for Chubb Personal Insurance's Underwriting. Most of the scenarios in this section come from strategies and real-world implementations by FICO customers in the Personal Lines Insurance industry.

FICO Sales and Marketing can provide further elaboration on selected content.



The subsections of this of this chapter cover the following scenarios for the application of BRMS within Personal Lines Underwriting.

1. BRMS and Underwriting for Personal Lines Insurance
2. New Focus for IT/IS: Revenue Generation
3. Making Revenue-Generating Decisions at Point of Sale
4. Getting a Better Handle on the Channel
5. Multiplying Profits and Markets through More Accurate, Innovative Pricing
6. Facilitating Profitable Renewals
7. Expanding the Market by Tightening the Market
8. Decision Management Is the Next Step

1.2.6 Business Rule Lifecycle Management

This section summarizes FICO's recommendations for a Rule Maintenance & Versioning Strategy for Chubb Personal Insurance (CPI). These recommendations are the result of a series of interviews that FICO conducted with CPI architects and business leaders, by which FICO gained an understanding of the issues and concerns that CPI has about maintaining and versioning business rules.

This section also incorporates findings from similar sessions with Chubb Specialty Insurance (CSI). The result is a unified approach to business rule lifecycle management that should be broadly applicable throughout the various insurance companies within Chubb & Son Insurance.

As rules become more broadly adopted throughout Chubb, various application areas are beginning to look for standard mechanisms to ensure not only the quality, but also the validity, traceability/auditability, history and versioning for their business rules as they are maintained and extended over time.

Processes are already in place within Chubb for source management, versioning, verification, testing, quality assurance, and publication of traditional source code. Chubb recognizes that the artifacts in a BRMS repository are different from traditional source code and sees obvious gaps where their traditional processes will not be adequate for BRMS. New processes are needed for BRMS.

Bottom line: business users, technical users, and managers all want to apply formal processes to ensure that only "good stuff" goes into Production.

Key Drivers:

- Desire for an automated lifecycle processes
 - To keep track of rule changes
 - To also keep track of model deployments
 - Parallel rule authoring efforts
 - Feedback loop into development
 - To better prepare for the future
- Step in the direction of full enterprise decision management

Important to look at this as business rule lifecycle management (BRLM) and not simply source code version management.

This section discusses the following aspects of a program for managing the lifecycles of business rule repository content:



1. *Blaze Advisor Product Features and Proper BRMS Structure and Design*—there are many features within Blaze Advisor that will assist Chubb in managing the content of their business rule repositories. While some of these features are being used, their use is not consistent and globally applied. There are other features which Chubb is either unaware of or has chosen not to use. This section discusses these features at a high level.
2. *FICO's Best Practices for Rule Versioning*—Blaze Advisor has been designed to work out-of-the-box with several of the leading commercial off-the-shelf (COTS) version management systems (VMS) being used in the industry. Blaze also provides its own Blaze Versioning System (BVS) for customers who do not have a version management system or are using an unsupported COTS VMS. This section discusses some FICO best practices for version management.
3. *FICO Best Practices for Business Rule Lifecycle Management (BRLM)*—This section provides an overview of FICO best practices for BRLM and use cases for typical BRLM scenarios. It is a predicate to the discussion of FICO's Lifecycle Management Service Framework discussed in Section 1.5.
4. *FICO Lifecycle Management Service Framework*—Lifecycle Management Service is a framework for managing the evolution of rules from creation to deployment in the Production environment. This service is delivered with a default implementation you can customize or replace with your own workflow tool.
5. *The "Maker – Checker" Approach*—FICO PS has developed an approach to managing some of the unfavorable side-effects from extensive use of management properties in large BRMS projects that are very volatile or undergo a high rate of maintenance. This section provides details on this approach.
6. *FICO Recommendations for CPI Rule Management*—this section provides a summary of the findings from the onsite sessions where FICO worked with Chubb CPI and CSI to understand Chubb's requirements and formulate suitable recommendations. It provides FICO's recommendation for an enterprise approach to Business Rule Lifecycle Management for the Chubb enterprise and then specific recommendations for CSI, CPI Print and CPI Underwriting.



2 Business Rule Management System (BRMS) Value Statement

For companies to be speed-competitive, the logic that drives business decisions can't be buried inside individual software systems. It must be visible, easily modified by nonprogrammers, and usable by any application and channel.

Business rules management is taking organizational computing by storm. That's because BRM liberates the logic governing operational decisions from individual applications, where it had been locked within programming code — similar to the way database management liberates data. Business rules management delivers substantial savings in application time to market and total cost of ownership.

Business rules can be anything your organization uses to make an operational decision. They might include enterprise, divisional, corporate and line of business policies, as well as calculations and formulas, risk thresholds and regulatory authorizations.

Rules are often expressed—in conversation, written text and software—as “if, then” statements: “If the applicant does not have a sufficient credit history, then pull a report from a debit bureau.”

Rules management solutions such as Blaze Advisor use a similar intuitive and readable syntax to specify and change rules in decisioning applications. Business users can therefore change how an application operates — immediately and without requiring IT help — just by modifying or replacing rules.

This section provides an executive-level overview of the value that a Business Rule Management System (RMS) brings to an enterprise.

2.1 Overview

This BRMS Value Statement is broken down into the following sections

1. BRMS Value to the Enterprise
2. Benefits vs. Added Complexities for the Business Community
3. Benefits vs. Added Complexities for the IT Community
4. ROI and Value throughout the BRMS Application Lifecycle
5. FICO Recommended Future Strategies for Decision Management at Chubb

2.2 BRMS Value to the Enterprise

Most organizations implement BRMS initially as a single project followed by a progression of successive tactical deployments. A first successful project typically causes broader interest within an organization, and BRMS is adapted more broadly. However, this adoption is typically done tactically on a project-by-project basis. Once an organization has adopted BRMS on a broad-enough scale, they typically begin to for centers of excellence around it and manage it like they would any other platform. The goal is to achieve value at the enterprise level.

Fortunately for the BRMS industry, the value that BRMS brings to an enterprise level is precisely the same as what it brings at the divisional, business unit, department, and even the individual application levels. Therefore, even though this discussion is focused at the enterprise level, these principles apply across the board at all levels of BRMS adoption.

This discussion is broken down into 4 main subject areas:

1. BRMS Value to the Application SDLC



2. BRMS Value to the Business
3. BRMS Value to IT
4. BRMS Value Assessment

The final section on BRMS Value Assessment will help Chubb assess for itself the value that it is receiving on individual projects and on the enterprise as a whole.

2.2.1 BRMS Value to the Application SDLC

Focusing the discussion simply on the values realized by the business and technical communities overlooks a broader concept of value that really just benefits everyone involved. For the purpose of this discussion, we will call the Value to the Application SDLC.

The table below summarizes the many challenges with the traditional approach to SDLC that impact both the technical and business communities. The benefits and values added by BRMS in mitigating the traditional challenges are contrasted in the right column.

Table 1—BRMS Value to the Application SDLC

Traditional Application Development Lifecycle	BRMS Decision Service Lifecycle
<ul style="list-style-type: none"> ✱ Rules written in COBOL or 3GL programming languages are buried deep in the system code of a single application. ✱ Rules can be executed only by the application they were originally written for; in the manner they were originally coded. ✱ Rule changes, which require reprogramming and can even involve reverse engineering, mean queuing up for IT help. ✱ New application development takes a long time because programmers must try to anticipate and code for every possible combination and interrelation of variables and conditions. Given today's complex business environment, that's just not possible anymore, so long development cycles are soon followed by extended enhancement cycles. 	<ul style="list-style-type: none"> ✱ Rules are written in a simple rule language and managed separately from system code. ✱ Rules can be used in many flexible ways to reach decisions that can be carried out by virtually any application or business process. ✱ Rule changes can be made quickly and inexpensively by business users without requiring help from IT. ✱ Rules management solutions enable new applications to be developed quickly, without the need to anticipate every possible decisioning scenario. As new situations and information emerge, enhancements are made by business users. New rules can be added at any time, without interrupting operations and with the rules engine sorting out how and when to execute them.



Traditional Application Development Lifecycle	BRMS Decision Service Lifecycle
<ul style="list-style-type: none"> Adding rules to legacy systems can be risky, given the fragility of some of these systems and the complexity of trying to make them perform functions they were never designed to support. 	<ul style="list-style-type: none"> Rules management services can be called by existing applications. This enables legacy applications to be reinvigorated with new capabilities without actually having to mess with them.
<ul style="list-style-type: none"> Extending decisioning across new channels usually requires building a duplicate application, from the ground up, to run in the new environment. 	<ul style="list-style-type: none"> Decisioning applications can be written once and deployed without modification across most any operating environment. IT can deploy and maintain a single application across multiple channels.
<ul style="list-style-type: none"> Development and deployment are often delayed because of difficulties translating complex business processes and requirements into code. There's too much opportunity for something to get lost between what the business people say they need and how the programmer interprets that in rules. It usually takes numerous revision cycles to get it right. 	<ul style="list-style-type: none"> Decisioning applications and services are developed usually by IT or a technically inclined business analyst—using a rules language that closely approximates the way business people talk about policies. Little interpretation is needed, so development goes smoothly. Developers can also let nontechnical business users write many of the rules themselves using graphical trees, tables or Web forms.

Next we will focus on the business and IT communities individually.

2.2.2 BRMS Value to the Business

The business community will perceive value from BRMS only in areas that directly affect the profitability of the organization. The top 4 benefits from which business communities see the most value from BRMS are:

- Make faster, more consistent business decisions* – Automation of decision processes minimizes the need for manual reviews, while ensuring consistency. The BRMS delivers the highest decision processing speeds, for any number of rules at any level of complexity.
- Change rules quickly with improved business agility* – Blaze Advisor is the most advanced, trusted and agile BRMS on the market. It is a complete tool for developing, simulating, optimizing and implementing rule changes faster than the competition, so you can respond quickly to changes in market dynamics.
- Reduce time to market* – BRMS helps you go to market quickly with application development that is 25% faster than conventional methods. Instead of coding rules in programming languages, new applications are developed using a rules language that closely approximates how business people think about their business, reducing the opportunity for errors or misunderstanding. Business users can change rules at any time, without disrupting operations and without IT assistance.
- Reduce total cost of ownership* – With an easy-to-use interface, natural language business rules and open architecture, BRMS helps you reduce operational costs. Reduce your initial



costs with quicker and more accurate development of applications. Reduce maintenance costs by up to 75% with business users doing most of the updates and enhancements.

2.2.3 BRMS Value to IT

The IT community will perceive value from BRMS in areas that increase their capabilities. Increasing the capabilities of IT makes them more productive and reduces the cost of resources. IT will find the most value in benefits that make them better, stronger, faster. The top 8 benefits from which IT communities see the most value from BRMS are:

- *Provides an integrated environment for development, authoring and testing* – Multiple methods can be used for rule creation and management, including decision trees, scorecards, decision tables, formula builder, graphical decision flows and customized templates.
- *Facilitates collaboration* – The Blaze Advisor system's enterprise rule repository enables developers to work in a coordinated manner as teams and leverage each other's work by sharing and reusing rules, rule sets, rule flows and object models. Additionally, project comparison of two projects in the same repository or across repositories identifies all the differences, with information on how they are different.
- *Enables users to add and manage an unlimited number of rules* – Groups of rules can be defined and maintained in easy-to-use constructs such as decision tables and decision trees.
- *Monitors decision performance* – Blaze Advisor monitors the business performance captured through user-defined events, providing the building blocks for strategy orchestration and champion/challenger strategies.
- *Simulates results* – The Decision Simulator module uses historical data to analyze the potential business impact of rules prior to moving them into production. Its wizards, pre-packaged templates and reports make it easy for business users to flexibly configure and run simulations for their specific needs. For more information, please refer to the FAQ.
- *Runs on an open system* – The Blaze Advisor system's open architecture enables ease of integration with any computing environment and accepts inputs from multiple databases, XML documents, Java objects, .NET/COM objects, and COBOL copybooks.
- *Executes rules based on the most advanced inference technology* – Blaze Advisor makes exclusive use of Rete III, the most advanced commercially available inference engine, which benchmarks more than 300% faster than competitive engines at the highest levels of complexity.
- *Provides complete rule tracking and version control* – Blaze Advisor keeps track of changes to rules over time, allowing viewing of past versions, audits of who made changes at what time, and return to previous versions if necessary. Security and locking mechanisms allow organizations to manage simultaneous user changes from multiple locations without conflict.

2.2.4 BRMS Value Assessment

The discussions above provided an extensive list of abstract benefits that organizations can expect to receive from BRMS. While most anyone would agree that all these points make good business sense and that any organization would receive value from such benefits, they are all abstract concepts and are therefore difficult to quantify and measure. If you cannot measure it, you cannot manage it.

FICO PS regularly conducts assessments and health checks on BRMS implementations and the SOW for those engagements often asks for an assessment of the value the client's organization realized from BRMS. So FICO created a simple scoring algorithm that produces a score that can be used to compare the value one organization realized against the broader industry. This score can be applied to an individual project, a business unit, or at the enterprise level.



2.2.4.1 BRMS Value Assessment Criteria

The following are the 5 criteria that will be used to determine the overall value realized from BRMS

- **Precision** – Select the optimal course of action in order to drive the maximum benefit from each specific customer, circumstance, or opportunity.
- **Consistency** – Make coordinated, repeatable decisions across business functions, across geography and time, and across personnel and situations.
- **Agility** – Respond faster to changes in customer preferences, competitive offerings, economic conditions, and regulatory oversight.
- **Speed** – Automate decisions that must otherwise be made manually. Accelerate the development of new applications.
- **Cost** – Save the cost of staff time needed to make operational decisions. Business rules can substantially reduce the cost of developing new applications, and the maintenance of applications by up to 75%.

2.2.4.2 BRMS Value Assessment Score

When we assess an individual project, a business unit, or even an enterprise, we assign a score from 0 to 2 to each of the 5 criteria, and then sum the total. The result is a score between 0 and 10.

Table 2—BRMS Value Assessment Scoring Criteria

Assessment of Value Realized	No	Somewhat	Yes
Did the Business realize value from this benefit of BRMS	0	0.5	1
Did IT realize value from the benefit of BRMS	0	0.5	1
Total Possible Score (per Criteria)			2

Here is an example from California State AA Auto Insurance:

Table 3—Example BRMS Value Assessment

Criteria	Decision Yield	Score
Precision	CSAA has been able to increase the accuracy of assessing risks on new and renewing policies with decision management strategies that include data-driven analytic models.	1
Consistency	CSAA has centralized the decision logic to make consistent underwriting decisions across systems without having to rely on individual underwriting expertise.	2
Agility	The BRMS solution allows flexibility and rapid deployment of underwriting business rules without having to depend on IT	2



FICO Recommendations Whitepaper for Chubb Business Rule COE

Speed	Manual decision-making has reduced by 80%. Automated decisions are in real time.	1
Cost	Through underwriting automation, CSAA has nullified the practice of continually hiring additional underwrites to keep up with the demand. Through BRMS and sharing services, the cost of maintaining the underwriting decisioning has reduced by 75%	2
Total Score:		8

As said above, the total score will be normalized between 0 and 10. That score can be used to gauge the value received by the project, organization, or enterprise relative to the BRMS industry in general.

Table 4—Quantified Description for BRMS Value Assessment Scores

Score	Value Assessment Relative to BRMS Industry
10	<i>A Perfect BRMS Implementation/Organization.</i> Receiving a 10 means that both Business and IT realized value on each of the 5 assessment criteria. While theoretically possible, in practice this score is not expected.
9	<i>Extraordinary Value.</i> This is an exceptional score and it indicates that the Business and IT communities are realizing more value than the methodology expects. Only the most mature BRMS organizations are capable of producing individual projects that score in this region.
8	<i>Very High Value.</i> This score puts the implementation or organization in the top 5%. This is the target score in this methodology. In a "methodology-perfect" BRMS implementation or an ideal adoption of BRMS for an organization or enterprise, it is not practical to expect every community to realize full value in all 5 assessment criteria. Assuming a realistic handful of 0.5 scores, the practical target score is an 8.
6-to-7	<i>High Value.</i> This score puts the implementation or organization in the top 25%. Regardless of whether this is a project or an organizational assessment, the organization is clearly realizing a substantial amount of value from BRMS.
5	<i>Normal Value.</i> The realized value certainly justified the investment in BRMS. There is room for improvement. However, no one will be regarding the project as being unsuccessful. This would be a realistic target score for a first endeavour in BRMS.
3-to-4	<i>Marginal Value.</i> The net return will likely offset the investment in BRMS. However, the project is not likely to be regarded by the organization as being successful.
0-to-2	<i>Little to No Value.</i> A fair assessment would conclude that the project or organization failed to realize any substantial value from BRMS.

March 31, 2011

FICO Confidential Page 24 of 259

Confidential

FED007129_0024



2.3 Benefits vs. Added Complexities for the Business Community

The goal of this section is to provide Chubb's business communities with a foundation for the open discussion of the relative benefits they can expect from BRMS so they can weigh those against the added complexities. The subsections cover:

1. BRMS Elevator Speech for Business
2. Business Challenges/Drivers for BRMS Capabilities
3. Benefits for the Business Communities
4. Added Complexities for the Business Communities

2.3.1 BRMS Elevator Speech for Business

A BRMS or Business Rule Management System is a software system used to define, deploy, execute, monitor and maintain the variety and complexity of decision logic that is used by operational systems within an organization or enterprise. This logic, also referred to as business rules, includes policies, requirements, and conditional statements that are used to determine the tactical actions that take place in applications and systems.

A BRMS includes, at minimum:

- A repository, allowing decision logic to be externalized from core application code
- Tools, allowing both technical developers and business experts to define and manage decision logic
- A runtime environment, allowing applications to invoke decision logic managed within the BRMS and execute it using a business rules engine

The top benefits of a BRMS include:

- Reduced or removed reliance on IT departments for changes in live systems. Although, QA and Rules testing would still be needed in any enterprise system.
- Increased control over implemented decision logic for compliance and better business management
- The ability to express decision logic with increased precision, using a business vocabulary syntax and graphical rule representations (decision tables, trees, scorecards and flows)
- Improved efficiency of processes through increased decision automation

2.3.2 Business Challenges/Drivers for BRMS Capabilities

The Business community's needs for Decision Management spring from the increasing complexity of making decisions:

- Decisions that once took days now have to be made at the speed of the transaction; such as while your customer is completing an online transaction.
- Business objectives used to be simpler and set at the local level—now those objectives involve trade-offs between risk, resource constraints, opportunity costs and other factors.
- The data available to make decisions has ballooned, but there are challenges to using all that data effectively.
- Compliance with more new regulations, stricter and more complex rules, shorter deadlines and greater consequences for non-compliance.



- Need to change customer treatment strategies more frequently and more rapidly to deal with competitive forces, environmental changes and changes in the customer base.
- Decisions that were once “owned” by a single group are “shared” by multiple departments, and have to be coordinated across channels and regions.
- Customers expect the same treatment regardless of channel, and they expect that the treatment is consistent with the value of their relationship.
- You used to be able to handle decisions requiring manual review processes—now the volume makes that impractical.

2.3.3 Benefits for the Business Communities

The main benefit that BRMS brings to the business communities is the elimination of many challenges or barriers that are inherent to traditional software implementations.

1. It is very expensive to operate within the confines of a traditional SDLC. The business community has to dedicate a large budget to operationalizing business decisions that are typically not overly complex.
2. While important to the integrity of an organization’s IT infrastructure, the rigor of a traditional SDLC is inherently burdensome on business. The time it takes to implement changes to business software is an impediment to speed and agility.
3. The traditional SDCL focuses on the implementation of individual business software applications that are purpose-built for specific needs. They are standalone. But business processes overlap. This leads not only to redundancy but also to inconsistency in the implementation of decisioning throughout an organization.
4. The communication of Business requirements in the traditional SDLC is akin to the children’s game of “Whisper Down the Lane” (a.k.a., “Pass the Message” or “Telephone”): The business communicates its functional requirements to a business analyst who analyzes them and communicates specifications to an architect who designs a solution and gives specifications to a developer who implements them... and then back to the business who tests the system and tells everyone they got it all wrong. This is a very imprecise approach to communicating requirements within an organization.

The above are more than just annoyances; these side effects of the traditional SDLC cost the business \$MM every year. Therefore, the real benefit to business is a reduction or elimination of losses that result from a business’s inability to be precise, consistent and agile in its decisioning. These include:

1. Lost revenue from imprecise decisions. Companies leave value on the table by making less targeted, less relevant decisions. For example, pricing insurance in one of three tiers when underwriting new customers will lead to lower profit per customer than a segmentation of 20 tiers—or even continuous pricing tailored to each customer.
2. Lost value through operational negation. Without systems that enforce consistency and connect decisions, businesses often make one set of decisions that negates the value created in another set of decisions. For example, the marketing department may attract customers that are unprofitable.
3. Lost share through falling behind the pace of change. When it takes weeks or months to change an offer, a pricing structure or a decision strategy, businesses cannot adapt fast enough to changes in consumer behavior or competitive offers. This leads to share erosion.

In addition to these financial benefits, other practical or tactical benefits of BRMS to the business community include:

1. Business Control—Business rules can be modified by business users, in a controlled, auditable manner that is cohesive and consistent across applications.



2. **Flexible and Consistent Workforce**—all company staff and all of the agents who act for a company treat customers consistently and according to the best practices identified.
3. **Learning Environment**—Business users and analysts can rapidly improve and evolve their decision logic by more quickly deploying new rules and models and by learning at a faster pace through continual feedback and champion/challenger strategy testing.

2.3.4 Added Complexities for the Business Communities

Nothing is free, so what are the costs for realizing the benefits described above?

- There is no technology purchase; Chubb owns an enterprise license to Blaze Advisor. So, what would normally be the first concern on the list is not a concern for Chubb.
- The business community will need to make time to participate in JAD sessions with business analysts from the IT community. These sessions will be frequent, non-trivial, and will last roughly the first 2/3 of the SDLC. However, the ROI that comes from taking a JAD approach to the implementation of BRMS is exceptionally high.
- The business community will need to undergo DRAW training.
- The business community will need to make a commitment to enhance their existing requirements analysis methodology with several key activities associated with Decision Requirements Analysis (DRA). This includes active participation in the following aspects of Decision Requirements Analysis:
 - Acting as subject-matter experts (SME) during the DRAW
 - Acting as subject-matter experts (SME) and providing elaboration on decisioning requirements during the Harvesting effort
 - Elaborating on RMA requirements
 - Elaborating on requirements for unit-test, acceptance-test, and regression-test suites
 - Approval/sign-off on decisioning requirements produced by DRA
- Collaboration with business analysts from IT in the specification of a rule maintenance and extension strategy. And, corollarily, participating in rule maintenance and extension as agreed upon in that strategy.

Business community participation (outside of business analysts, who are IT) on mid-scale projects typically amounts to approximately 1 FTE over the first 2/3 of the SDLC. Involvement will be higher to start and then taper off to cycles of moderate involvement separated by periods of light involvement.

2.4 Benefits vs. Added Complexities for the IT Community

The goal of this section is to provide Chubb's IT communities with a foundation for the open discussion of the relative benefits they can expect from BRMS so they can weigh those against the added complexities. The subsections cover

1. BRMS Elevator Speech for IT
2. Technology Challenges/Drivers for BRMS Technologies
3. Benefits for the IT Communities
4. Added Complexities for the IT Communities



2.4.1 BRMS Elevator Speech for IT

A BRMS provides a complete rules environment that allows complex business decisions to be abstracted from hard-coded applications, Web pages, straight-through processes, or composite applications. In doing so it increases the stability and code quality of the higher-level object, ensures the consistency of application of business policy across the enterprise, and enables greater responsiveness to change. It provides visibility and audit of automated decisions in a manner that is meaningful to business users and IT staff alike.

A BRMS provides a high-performance runtime environment for .NET, Java, Service Oriented Architecture (SOA), or legacy applications and features strong change management through its rules repository. It can form part of a more comprehensive enterprise-level Decision Management (DM) strategy, or simply satisfy the tactical need for greater manageability of variable business logic.

2.4.2 Technology Challenges/Drivers for BRMS Technologies

The traditional approach to application architecture is to embed all the decision-making logic within hard-coded applications. At face value this provides the simplest and lowest-overhead architecture, but in fact leads to a number of serious disadvantages:

- Changes to the logic can only be carried out by a programmer, and the consequent new version of the application has to pass through all of the testing and acceptance procedures before it can be deployed, owing to a greatly prolonged change cycle.
- Where the same logic is required in multiple applications there is a real danger that changes will not be made consistently, leading to variations on the anticipated system behavior. This is a particular concern where the same decision logic is required to be implemented within different application styles – e.g., .NET, Java, and COBOL.
- It is very complex to audit the decision-making process to determine why any particular transaction took the course it did.
- The code cannot be presented to business decision makers in a form that enables them to check the logic is as required.
- Very complex decisions are notoriously difficult to code in an efficient manner, and the actual runtime performance may be inadequate. Organizations often resort to manual decision making even though it would be possible to define the rules.

The solution increasingly being adopted is to deploy a Business Rules Management System (BRMS) as a single point for managing all decision logic. Chubb has standardized on the Blaze Advisor BRMS product from FICO.

2.4.3 Benefits for the IT Communities

In general, the one benefit that IT looks for in any technology or methodology is “does it make us more productive?” IT needs to implement solutions and maintain systems/applications as efficiently and effectively as possible. This section will examine the benefits of BRMS from that perspective.

The aim of Blaze Advisor is to provide a single point of decision management at the enterprise level. The primary features that enable this strategy are:

- The ability to create rules and rule sets to resolve complex decisions, while presenting the logic in an intuitive manner that can be understood and modified by business or IT staff.
- Development Productivity—Business rules are built and maintained once, not replicated in fragmented systems.
- Deployment across a heterogeneous environment, including deployment to client-side Web forms.



- Direct integration of analytics through scorecards and other visualization mechanisms.
- Analytic Consistency—Analysts can use the same methodology to build models for different application areas.
- High-performance runtime environment with a choice of decision algorithms.
- Testing and change management capabilities.
- Real-time Deployment—Models can be deployed into the production environment immediately after development.
- Production Fortified—FICO's business rule execution environment is the cornerstone of a production environment that has a seven-year history of zero production downtime.

Although most effectively deployed at the enterprise level, the same features make Blaze Advisor appropriate for tactical requirements where it is required to manage complex decisions in discrete projects.

2.4.3.1 Effective Decision-Authoring Methodology

By providing two authoring environments, Blaze gives IT the ability to create rules and rule sets to resolve complex decisions in the IDE, while presenting the logic in an intuitive manner that can be understood and modified by business or IT staff in the RMA.

Resolving a complex decisioning problem into a well-designed ruleset of discrete atomic rules has an effect of simplifying the decisioning logic. As a result, the rule (analogous to code) volume is smaller and much less complex. They are more discrete and encapsulated, which makes task assignments more manageable.

The template approach improves productivity substantially. Once a template is designed, multiple instances of it can be generated and configured quickly. The configuration of a rule is also much less technical than authoring source code. Therefore, non-technical analysts are able to join the development team and assist in the writing of business rules using the RMA.

The RMA also provides a great metaphor for the business community to review the rules and sign-off on them before promoting the rules out of the DEV environment. The RMA can also be integrated with test harnesses and simulation frameworks that will allow the business community to actually validate the correctness of the decisioning logic before the rules are released out of DEV.

Development Productivity—Business rules can be shared across multiple BRMS projects. Therefore, common or shared rules are built and maintained only once, they are not replicated in fragmented systems like application code.

2.4.3.2 High-Performance Runtime Execution Environment

Production Fortified—FICO's business rule execution environment is the cornerstone of production environments at many FICO customers that have 7+ years of history with zero production downtime. The runtime technology is reliable.

Performance—FICO owns the IP rights to the fastest inference algorithm in the industry. FICO has customers in the banking industry who use Blaze advisor to process every credit/debit card swipe at every ATM and point of sale for fraud. When needed, a Blaze BRMS solution can be designed to provide sub-millisecond transaction processing speed.

2.4.3.3 Deployment across Heterogeneous Environments

The runtime environment for Blaze Advisor supports Windows and Java on major Unix and Linux environments (including mainframe Java). While it can execute within a simple Java Virtual Machine (JVM), it is best deployed on a JEE application server, where it can use the distribution and fault-tolerance features provided. For legacy COBOL applications, Blaze Advisor can generate compiled



code instead of using a runtime rules execution engine. A single rule set can be deployed to any number of runtime environments to place the intelligence physically close to the point at which it is needed. Because Blaze Advisor executes on the same platform as the calling application (or process) the communication costs and performance overheads are minimized. The development environment requires either Windows or Solaris. The repository can be deployed on the host file system, in which case it has no third-party prerequisites, or alternatively on a Relational Database, or hosted within a change management system or Lightweight Directory Access Protocol (LDAP) directory.

2.4.3.4 Direct Integration of Analytics

Blaze Advisor provides one platform for deploying all decisioning capabilities within a BRMS application. It supports direct integration of analytics through scorecards and other visualization mechanisms.

Analytic Consistency—Analysts can use the same methodology to build models for different application areas.

Real-time Deployment—Models can be deployed into the production environment immediately after development.

2.4.4 Added Complexities for the IT Communities

Asserting BRMS adds complexities for the IT community is unfair. It is true that BRMS approaches a solution in a way that is different than traditional software development. However, there are no grounds for any assertion that it is more complex. In fact, simplification of decisioning logic is one of the benefits of BRMS. Treating decisioning artifacts (rules, rulesets, etc.) as discrete entities simplifies business rule lifecycle management. Therefore, FICO sees this Chubb's perception of added complexities as more of a maturity, experience, or education issue than a technology/methodology issue.

There is, however, one aspect of the methodology that is more complex for BRMS than for traditional development. The net-net is still a substantial simplification because there are dozens of aspects that are simplified and only one aspect that is complicated. This aspect is version management.

1. Versioning is different for rules than source code. Source code is managed at the file level. The granularity of versioning for BRMS is at the repository entity level.
2. Every repository entity can have its own lifecycle. Therefore version management is broader than simple file locking and check-out/check-in.
3. Lifecycle management is more complex than what is supported by the COTS version management systems. Therefore, multiple tools and methodologies are needed.
4. Blaze Advisor provides a wide variety of tools (like meta-data, publish/release, filters, queries, etc.) to support lifecycle management.

Details on version management are provided in a later chapter of this whitepaper.

2.5 ROI and Value throughout the BRMS Application Lifecycle

The value provided by BRMS is different at various stages of the application lifecycle. The application lifecycle is much broader than the software development lifecycle (SDLC) that is the focus of many IT professionals. The application lifecycle begins when someone first says "wouldn't it be great if we had a software application that does _____." It continues through to the end of sunseting.

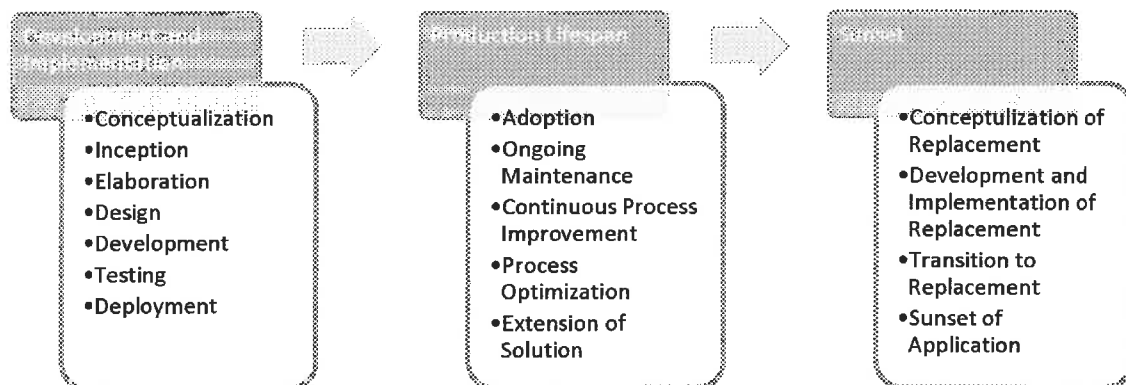


Figure 3—Software Application Lifecycle

Somebody once poked fun at the application lifecycle by asserting: “We spend two years talking about an application, followed by two years building it, to get at most two years of productive use out of it, follow by 2 years of trying to change it, two years talking about replacing it, two years developing its replacement, two years adopting the replacement, followed by ten years trying to get the heck off of it.” The anecdote is intentionally extreme. The point being made is that the application lifecycle is quite long and the Business is not receiving much value from the application throughout the vast majority of its lifetime. That is a very common situation that just about everyone in the IT and business application software.

This section looks at the value that BRMS brings to an organization throughout this long application lifecycle

2.5.1 BRMS Industry Assertions about Reducing the Cost of Development

There is a common assertion made by all products in the market that BRMS technology will drive down the cost of development because BRMS will allow you to develop the application more quickly, more accurately, and using business resources. Some businesses have estimated this reduction in initial cost to be as high as 80%.

This is not an untrue statement, or an unrealistic assertion. Unfortunately, through the law of unintended consequences, it can be a very damaging one. It gives the business community the impression that by purchasing BRMS they will be able to development and implement an application 80% faster. It is not uncommon to see Business communities interpret this as: a SDLC that would normally take 10 months will now only take 2 months.

The assertion is being made about the costs of implementing the business rules. BRMS give huge uplifts in productivity when it comes to implementing business rules over coding them in a natural language like Java or C#, or developing table look-up solutions using a RDBMS. However, the business rules are a just a part of a larger broader implementation. The other aspects of the application will not be impacted by the BRMS.

Realistically, organizations can expect to see an overall initial cost reduction of between 5% and 10%; which is a huge improvement and a substantial savings. This in itself is often a sufficient amount value to offset the investment into BRMS technology and training.

2.5.2 Reinvest the Value from Faster Development into Better Development

There are remarkably different perceptions on the BRMS value proposition between the BRMS community that sells the technology and the BRMS community that implements the technology.



Industry marketing materials often invite business to recognize the value and savings from faster development during development. This certainly makes sense from a budget and corporate ledger perspective. However, it is a short sighted perspective and is actually a missed opportunity.

Review the benefits and characteristics of BRMS listed in Section 2.2 above. The majority of the benefits of BRMS come during the mid-life of an application's lifecycle. Businesses are reporting savings of up to 75% in maintenance cost. Businesses are handling their own updates and enhancements using the RMA. Applications are staying in production longer. End-user satisfaction is higher. Businesses are more agile. And these figures do not take into consideration the absence of abstract lost-opportunity costs because businesses are now capable of making minor changes to decisioning that they would not have been able to justify under the traditional model.

The perception on implementation cost benefits from the BRMS community that has experience implementing the technology is to infest the savings from faster development into better development. Use the time you save coding the rules to do a better job building the thing that will pay dividends during the mid-lifecycle. Some suggestions include:

1. More comprehensive Decision Requirements Analysis (DRA)
2. More thorough unit test suites
3. Enhanced usability features for the RMA
4. RMA with an integrated test harness
5. RMA with an integrated simulation framework
6. More comprehensive traceability and validation tools
7. More comprehensive Business Rule Lifecycle Management (BRLM) tools
8. Structure for reuse and governance
9. Document, document, document
10. Enhance the maturity of the COE and the corporate BRMS Methodology

By choosing to forego short-term recognition of savings investing in enhancement such as those listed above, organizations will realize the greatest value and highest ROI from their use of BRMS.

2.6 FICO Recommended Future Strategies for Decision Management at Chubb

Business Rule Management is part of a broader set of technologies and methodologies known as Decision Management. Chubb is just starting to tap the potential of Decision Management to unlock value for businesses and consumers. Organizations that embrace Decision Management will find it easier to attract and retain customers, employees and operate more profitably. Organizations that get there first can count on a true competitive edge.

As Chubb achieves maturity and embrace the full potential of BRMS, the next step to consider is broadening their Decision Management capabilities. The following are some examples of where FICO sees opportunities for Decision Management in Chubb.

2.6.1 Get Rid of the Silos

Standardizing on a single decision management system will allow managers to be able to make one change to a rule base and have it impact decisions throughout the business. This will encourage faster decisions and nimbler rules management, and will reduce costs while ensuring the company's best thinking directs every interaction. In an age of increasing regulation, the need for large, diverse companies to turn on a dime will increase the demand for a Decision Management backbone that matches precision and consistency with agility.



2.6.2 Bring Equations to the Planning Table

Using analytics as a business planning tool will help remove uncertainty from product launches, marketing campaigns and other highly complex operations. FICO recommends that Chubb should start looking out FICO's decision models—analytics that calculate the connections between decisions, results and objectives—as tools to stress-test risk management strategies under changing economic conditions and develop the most effective marketing mix for major promotional campaigns.

2.6.3 All-points fraud protection

Fraudsters quickly find and exploit the weakest links in any financial organization, including insurance companies. They rely on the fact that most businesses, even those with significant investments in fraud prevention and detection, have an uncoordinated approach to dealing with fraud. FICO has developed enterprise fraud management systems that protect multiple channels or products using the same rules management system, profiling and analytics. Information, alerts and analyses flow throughout the enterprise, creating a fraud barrier with maximum strength at all touch points.



3 BRMS in Personal Line Insurance Industry Overview

This section provides an executive-level overview of BRMS usage in the Personal Lines Insurance Industry. The subsections include current trends for BRMS in the Insurance Industry, historical successes, historical challenges, and guidelines for conducting suitability analysis on projects.

3.1 Overview

The subsections of this chapter cover the following subject-matter as it pertains to the application of BRMS in Personal Lines Insurance:

1. Current Trends for BRMS in the Insurance Industry
2. Historical Successes—Things That Make BRMS Successful
 - a. Select the Right Application
 - b. Follow a Methodology
 - c. Document, Document, Document
 - d. Manage Traceability
 - e. Manage Business Rule Quality
 - f. Choose the Right Metaphor
 - g. Verify the Business Rules
 - h. Validate Business Rules
 - i. Simulate the Business Impact
 - j. Structure for Reuse and Governance
 - k. Operationalize Analytics and Improve Decisions
3. Historical Challenges—Things That Add Risk to BRMS Projects
 - a. Challenging Domains
 - b. Challenging Business Requirements
4. BRMS Application Suitability—Guidelines for Selecting App for BRMS
 - a. Assessment of Volatility and Complexity of Decisioning
 - b. Indicators and Contraindications for BRMS
 - c. Sources of Knowledge
 - d. Types of Decisioning
 - e. Types of Functional Requirements

3.2 Current Trends for BRMS in the Insurance Industry

This section provides a synopsis of Blaze Advisor applications within the Personal Lines Insurance industry over recent years. It begins with a brief overview (case study) on various real-world

applications of Blaze Advisor. That is then followed by a detailed whitepaper on the applicability of BRMS to underwriting for personal lines insurance.

3.2.1 Case Studies of BRMS at Some Major Personal Lines Insurance Carriers



3.2.1.1 Point of Sale Quotations and New Business Decisions



The Hartford Insurance Group is moving all of its decision logic onto FICOTM Blaze Advisor®, the world's leading business rules management solution. Blaze Advisor enables The Hartford to combine its own predictive models with rules and other decisioning elements—all without programming—and rapidly deploy the automated decisioning processes to multiple frontline interfaces, including third-party agency management applications (with automatic transformation to/from XML) and a customer self-serve website. Currently new BRM-based processes are being used for real-time POS quotations and new business; soon they'll be extended to renewals as well.

3.2.1.2 Better Decisions at Five Times the Volume



A Large Personal Lines Carrier has been able to increase the volume of new business it's writing by four to five times while improving the objectivity, thoroughness and consistency of its decision-making processes. They did it by incorporating credit-based insurance scores as a component in custom models that also look at other factors to predict loss risk. The models, built by FICO, are deployed in the carrier's mainframe-based underwriting system, which delivers web-based POS decisions to a network of independent distributors.



The senior executive in charge has this to say:

"If a company wants to grow rapidly and have confidence that the new business selection is solid, the only way it can be done is with analytics."

3.2.1.3 Dynamically Updated Forms



Republic Indemnity, a division of Great American is using business rules management for POS quotations and new business. The company's real-time solutions include dynamic forms, which it built with the assistance of FICO. Nontechnical managers can now make changes to decisioning rules themselves—in one central place, with no programming required. The decision logic in the browser-based forms used by Republic's agents are then automatically updated. Now there is no lag time between changing policies and deploying them, and consistency is guaranteed.

3.2.1.4 Real-Time Underwriting



Kemper Insurance replaced its mainframe-based, rules driven underwriting system with a Decision Management solution that provides more flexibility to modify underwriting guidelines and apply them consistently across channels in real time. The solution also incorporates FICO predictive models for forecasting loss ratio. As a result, Kemper has attained its goal for reducing underwriting losses along with an 8-point drop in its combined loss ratio. Underwriting staff have also been freed up to spend more time focusing on book management and agency performance.



Kingsway Financial had merged five companies under its banner, each with its own underwriting system. Taking a Decision Management approach, Kingsway now maintains underwriting rules for all five systems in a single, central repository. Business managers for each company can access the repository to add or change their own rules using their own unique GUI. At the same time, corporate-wide rules can be implemented once and applied to all.

Other carriers who have adopted FICO solutions for underwriting:

- | | | |
|---------------------|--------------------------------------|--|
| ➤ AAA Michigan | ➤ Guardian Life Insurance Company | ➤ Kingsway Financial |
| ➤ Acuity | ➤ Harleysville Insurance | ➤ Liberty International Holdings Company |
| ➤ AIG | ➤ The Hartford | ➤ Nationwide Insurance |
| ➤ AXA Insurance Co. | ➤ John Hancock Mutual Life Insurance | |
| ➤ Erie Insurance | | |
| ➤ Fiserv AIS | | |



3.2.1.5 Consistency in Claims Processing



Prudential Life Insurance is using business rules management to handle all aspects of death benefits claims handling. From a single interface, the claims department can control call scripting, data acquisition and data validation—and make changes in any area in just minutes. The benefits include lower costs and training time, improved regulatory compliance and customer and employee satisfaction.

3.2.1.6 Improvement in Market Share

While business rules can be combined with sophisticated analytics to perform precise market segmentation and targeting, they can also be used in a more prosaic way to attract and engage potential new customers.



GEICO, for example, found that 9 out of 10 visitors to its self-service website were abandoning the quote process before completion. Using business rules, the site now intelligently directs the online quote process, asking only relevant questions in the context of driver risk class and state regulations. Today 50% of visitors stay long enough to receive their quote.



Samsung Fire & Marine Insurance wanted to attract a younger generation of insurance buyers. They used business rules management to create an online buying guide that makes personalized product recommendations to youthful customers based on their profiles and on-site behavior. Rules also analyze customer interests in the company's various products and provide data to marketing teams. These nontechnical marketing folks can change how the site interacts with users and what products it recommends by changing the rules, all by themselves, on a day-to-day basis if necessary.

3.3 Historical Successes—Things That Make BRMS Successful

This section provides an overview of application areas, implementation techniques, and best practices that has made Blaze Advisor successful in Personal Lines Insurance applications.

When automating decisions in areas of Personal Line Insurance from underwriting to claims management to fraud, insurers need a platform for defining the policies and regulations that drive those decisions. A modern business rules management system (BRMS), such as the FICO™ Blaze Advisor® system, is ideal. However, it is not enough to pick the right technology. The technology must be applied in the right way.

Based on decades of experience developing decision management applications, FICO has developed 11 steps to help our Insurance customers make the most of business rules. This paper shares these secrets and addresses:

- Picking the right application and development approach
- Writing rules as effectively as possible



- Ensuring the right rules are written and that they have the expected impact
- Operationalizing analytics and improving decisions

3.3.1 Select the Right Application

Business rules are an effective technology that can deliver a strong return on investment. However, not every application area is equally suitable for automation using business rules.

Blaze Advisor is designed to automate decisions. Clearly, then, the system or business process being automated must involve a decision or decisions to be a suitable application. And these decisions should be repeatable and occur reasonably often—at a moderate to high volume. Decisions that are always made differently or only made occasionally are not likely to be good candidates.

If you have determined that decisions are important to your implementation, there are a number of characteristics to look for. The decisions might:

- Involve numerous rules, such as property damage claims or medical bill review.
- Have rules that change frequently, such as marketing promotions.
- Require quick changes to meet short time-to-market windows, such as pricing in very competitive markets.
- Have rules that embody business domain knowledge best maintained by business people, such as medical rules or underwriting rules.
- Be complex or involve rules that interact in complex ways, such as life insurance underwriting.
- Require multiple levels of reasoning, such as risk analysis, underwriting and identifying allowed configurations.

Good candidates have at least one of these characteristics and the best candidates have several. Sometimes a decision exhibits a particular characteristic so strongly it is clear that the application is a good candidate. Other times the combination of several characteristics makes a BRMS like Blaze Advisor worth considering.

It is often clear that some or all of these characteristics will be true early in the specification of requirements. In addition, you should consider a BRMS if during requirements you hear words like “consequently,” “therefore,” “I can only do this when...,” “Unless this is true I can’t...” and “This is a prerequisite for the next step.”

Examples of Suitable Decision Areas in Personal Lines Insurance

- | | |
|-------------------------------|----------------------------------|
| • Underwriting | • Marketing Cross-Sell / Upsell |
| • Claims Processing | • Group Enrollment |
| • Credit Risk Scoring | • Commission Calculations |
| • Regulatory Compliance | • Phone Usage Analysis |
| • Product Configuration | • Pricing |
| • Product Recommendation | • Intelligent Call Routing (CRM) |
| • Business Process Automation | • Fee Calculations |
| • Benefits Analysis | • Eligibility |
| • Fraud Alerts | ... and many more |

3.3.2 Follow a Methodology

Just because you have decided to use business rules and a BRMS like Blaze Advisor does not mean you can forget all your systems development best practices. In particular, it is still important to follow a well thought out methodology. Business rules work well with methodologies from SCRUM and XP to the Rational Unified Process.

FICO's Project Delivery Methodology is based on the Rational Unified Process, which is a widely recognized delivery methodology used by an increasing number of our clients and partners. The methodology includes an iterative approach where risks are identified at the outset. To ensure project success, these risks are addressed early and often within the project lifecycle. The figure below gives an overview of the various phases typically used and the disciplines applied in each, though this is customized for each project.

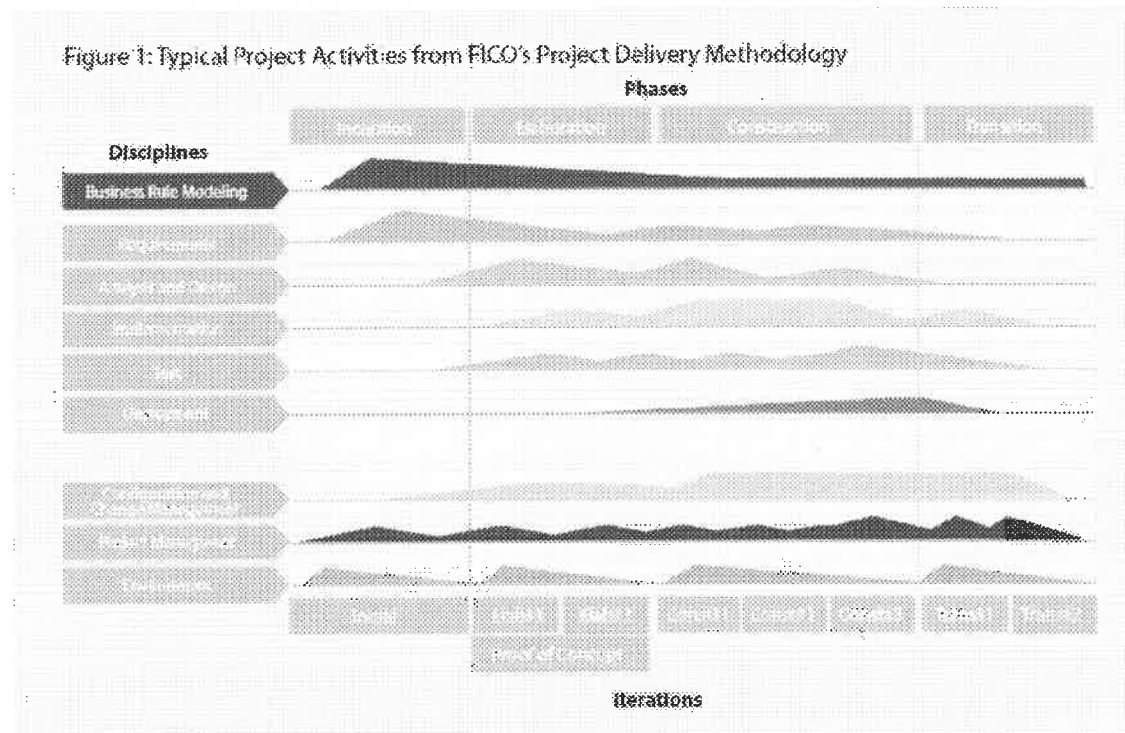


Figure 4—Typical Project Activities from FICO's Project Delivery Methodology

Integrating the activities required to discover, document, develop and maintain business rules with your preferred systems development methodology maximizes the likelihood that business rules will be a success in your organization. It is equally dangerous to ignore either your existing methodology or the need to make changes to it to support business rules.

3.3.3 Document, Document, Document

Just as using a BRMS doesn't eliminate the need for a methodology, it also doesn't eliminate the importance of documenting your requirements. The first step is to document the business processes you're working with, using a business process map, for instance. Then you'll drill down into the details of your use cases. Use cases contain decisions—not business rules—and you need to identify all the decision points within your use cases. You will find decisions that appear in multiple use cases as well



as dependencies between decisions. Identifying the decisions explicitly will help you manage this information.

When drilling into the decisions, you need to document the business rules that make those decisions, the terms that are used in those rules and other rule metadata, such as the rule's source. Writing effective, maintainable rules will be much easier if you document what your terms and rules are, where they come from and how you plan to use them. You can document the purpose of a rule within the development environment, or use a third party product like RuleGuide or RuleXpress. You can also use a Word document or Excel spreadsheet.

One approach, the one FICO uses, is to develop a Decision Set Template Structure. This spreadsheet is a useful thinking tool to document and structure the rules you will need. Such a templated approach is more rigorous than just writing natural language rules, but it is still readable by a business person and it helps you organize rules into conceptually related groups. In general, the larger the number of rules associated with a decision, the greater the benefit of this approach.

Figure 2. Rule Specification Template

Conditions					Conclusions						
If	and	and	and	and	then						
Gender	Age	State of Residence	Marital Status	Occupation	Risk	Severity	RiskID	Start	End	Last	Comments
M	>26	OH	M		med	1	Risk01	8/20/2006	RJB	11/21/2006	
M	<28				high	2	Risk02	8/26/2006	RJB	11/15/2006	
F	>26		M	Type 7	med	3	Risk03	8/20/2006	SAM	11/15/2006	
M	>50		NM		med	4	Risk04	9/4/2006	LO	11/21/2006	
F	>36 & <72				low	2	Risk05	8/20/2006	LO	8/21/2006	
F	>60		NM	Type 3	med	5	Risk06	10/1/2006	CVT	11/21/2006	
M	>62	CA	NM		low	4	Risk07	8/20/2006	RMB	11/21/2006	
F	>48 & <62		M		med	4	Risk07	11/17/2006	JFK	11/17/2006	

Figure 6—Rule Specification Template

3.3.4 Manage Traceability

While Blaze Advisor makes it easy to change the rules in your system, traceability to the original source helps ensure you make the right change. This traceability needs to be documented and maintained. Thanks to the Blaze Advisor system's extensible repository, you can record any source information you need—the law it came from, the business unit that defined it, owners and approvers, and more. These Management Properties can be defined for individual rules, rule sets or any artifact in Blaze Advisor. Once stored, these properties are managed and versioned by Blaze Advisor and the powerful query capability provides excellent impact analysis, allowing you to find every artifact used to implement a particular regulation, for instance, or driven by the needs of a particular department.

Business rule updates are driven by changes in the real world. Good management of traceability will help you find the right rules and artifacts to update to meet changing business needs.

3.3.5 Manage Business Rule Quality

There are many measures of business rule quality, but two of the most important are that business rules must be concise and atomic. Ensuring that business rules are both concise and atomic makes it easier to confirm that their behavior is what the business needs and, crucially, makes it possible to modify the rules easily over time.



Concise business rules only mention the concepts that are absolutely necessary to decide what action should be taken or otherwise draw a conclusion. While a business rule should include all the conditions that govern its applicability, it should not specify unnecessary conditions, which would artificially limit the applicability of the rule. Consider, for example, the following rule:

If applicant's gender is "Male" and applicant has a Criminal Record and applicant's number of accidents is greater than or equal to 2 and applicant's age is less than 25 then set applicant's risk to HIGH

Are all four conditions needed for this rule? For instance, would an applicant be high risk if they were a male under 25 years who had two or more accidents, even if they did *not* have a criminal record? If so, the criminal record check is redundant and should be removed from the rule to make it as concise as possible.

Atomic business rules keep the concepts addressed by the rule as simple as possible. This means limiting the conditions and actions of the rule to one concept or activity wherever possible. An atomic business rule should be focused on just one concept or outcome. Consider this business rule that has two outcomes:

*If the Applicant is Non-Smoker
then the Applicant qualifies for a 10% discount
and the Applicant qualifies for a free duffel bag*

Two completely different business changes would require us to change this rule—any change to the discount policy or to the duffel bag promotion. If we broke this rule into two atomic business rules we would get:

*If the Applicant is Non-Smoker
then the Applicant qualifies for a 10% discount*

*If the Applicant is Non-Smoker
then the Applicant qualifies for a free duffel bag*

Making this business rule into two separate *atomic* business rules allows these concepts to be managed more effectively because the conditions can be changed independently for the two outcomes. For instance, when written atomically it would be easy to modify them if the criteria for free shipping changed:

*If the Applicant is Non-Smoker
then the Applicant qualifies for a 10% discount*

*If the Applicant is Non-Smoker and the Applicant is a Student
then the Applicant qualifies for a free duffel bag*

Similarly if a business rule contains multiple sets of conditions where either set can trigger the rule then the rule is not atomic. For instance:

*If Age is less than 18 or State is not CA
then reject the Application*

In this case we would have to edit the business rule if we wanted to change our response to either set of conditions. Atomic business rules would separate these conditions out, making the business rules independent and easier to maintain separately:

*If Age is less than 18
then reject the Application*

*If State is not CA
then reject the Application*

Representing them atomically in this way allows us, for instance, to easily change the policy so that applicants who are under 18 but above 16 are not rejected, but instead charged a 120% surcharge in premiums.

*If Age is less than 16
then reject the Application*

*If Age is greater than or equal to 16 and Age is less than 18
then Set Premium Surcharge to 120%*

*If State is not CA
then reject the Application*

Business rules should be developed to be concise and atomic to ensure they are easy to check and easy to modify in response to changing business conditions.

3.3.6 Choose the Right Metaphor

When editing business rules, it is critical to consider how the rules are going to be authored. One of the first considerations is to establish how the rule will be edited—what elements can be changed and in what ways.

In Figure 6 below, you might start with a simple text rule (1). To help a business user edit this rule safely and easily, you could establish edit styles—let them select the state and enter a value, for instance (2). Over time you might decide that you also want to be able to specify exception rules, and could add an ability to choose between “does” and “does not” live in the specified state. You might also allow different kinds of comparison (not just less than) and allow rules to specify Accept or Reject, not just Reject (3). Ultimately you might allow the rule to be applied only to one of your defined customer segments (4) or give a user complete flexibility to create and edit rules (5). Support for flexible degrees of authoring should be part of the BRMS so you can develop this flexibility over time.

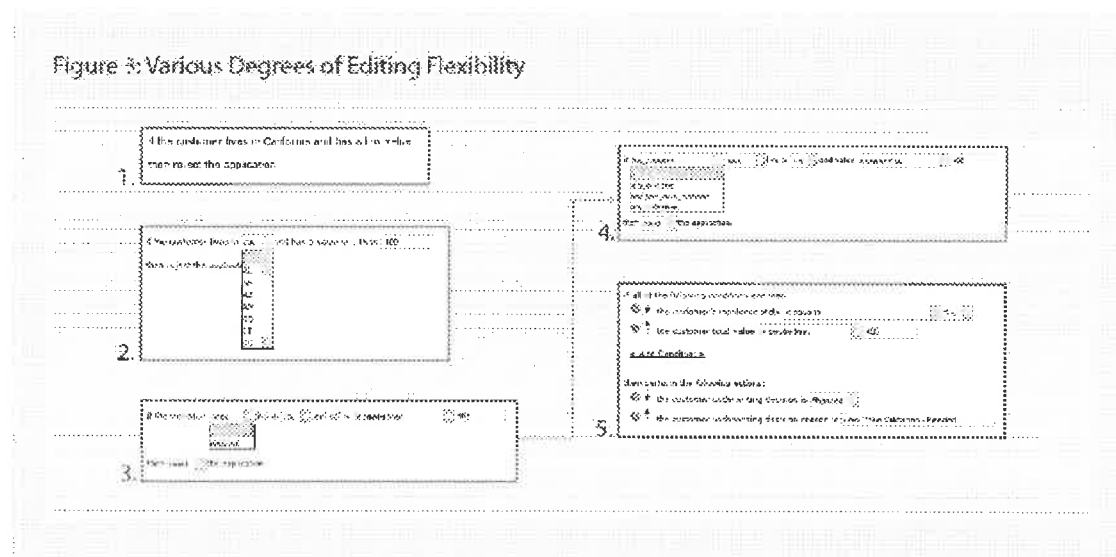


Figure 6—Various Degrees of Editing Flexibility

While the “if ... then” style is the default or “classic” style for a business rule, it is not the only style. Many situations call for sets of rules to be written. Instead of writing your rules one at a time, you can write them as a set in a decision table, such as the one shown in Figure 7. Decision tables are



particularly effective with business rules managed in tables, such as fee tables or actuarial tables. They can dramatically reduce the number of independent rules required. Decision tables also offer easy business rule definition and maintenance and can be used when a table is a familiar representation for business users.

Figure 4: A Decision Table

Reason for the Claim	Gross Medical Claim	Company	Pay the Amount
Routine_Office_Visit	0 < <= 150	National	NationalCompanyRateTable
Routine_Office_Visit	>= 150	National	Max Amt \$50 Rate NationalCompanyRateTable
Routine_Office_Visit	>= 150	Worldwide	Max Amt \$50 Rate WorldwideCompanyRateTable
Followup_Visit	0 < <= 200	National	NationalCompanyRateTable
Followup_Visit	0 < <= 200	Worldwide	WorldwideCompanyRateTable
Followup_Visit	>= 200	National	Max Amt \$200 Rate NationalCompanyRateTable

Figure 7—A Decision Table

In a decision table, each cell represents a business rule. In the example, the top left cell represents the rule:

*If Reason for the Claim is Routine_Office_Visit
and Gross Medical Claim is between 0 & 150
and Company is National
then use National Company Rate Table to Pay Claim*

Decision tables may not be appropriate for all sets of rules, however. If the rule set is very sparse or if the condition action pairs are not symmetrical, then a decision tree as that shown in Figure 8 on the following page is more appropriate. It adds maintainability by making such sparse asymmetric logic evident to the business user.

Decision trees have conditions on their branches and actions on their leaves. In the example, for instance, the bottom path through the decision tree represents the business rule:

*If Plan is FPO and Service is Emergency
and Location is Foreign
then Review Required to Pay Claim*

In all cases your BRMS should allow you to use business terminology to define conditions and actions. It should also provide a graphical point-and-click environment for both development and maintenance of

these metaphors. Blaze Advisor does both while also providing a wizard-based set-up process and the ability to embed these metaphors in browser-based thin client rule maintenance applications. Blaze Advisor also provides another metaphor, score models, but these are designed to implement predictive analytic models and will be discussed in that context. A good BRMS, and a good project methodology, will make it easier to ensure that the right business rules are being written, and written the right way.

Figure 5: A Decision Tree

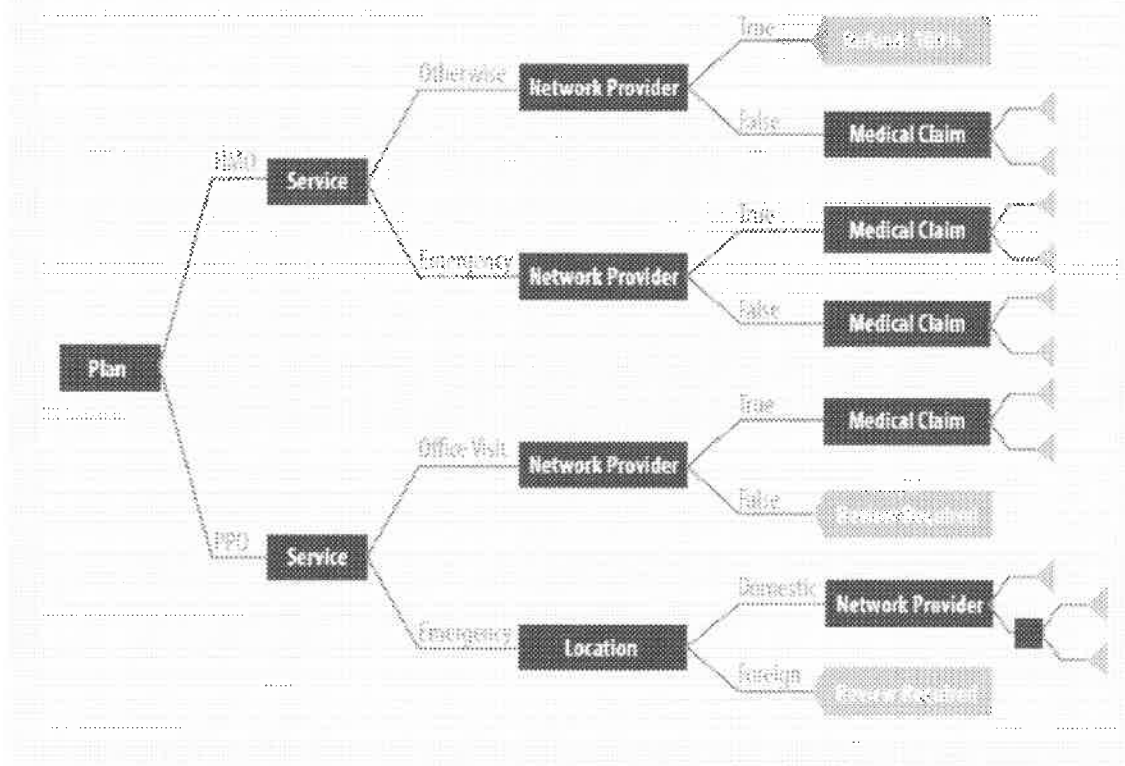


Figure 6—A Decision Tree

3.3.7 Verify the Business Rules

Having written the business rules, or after making changes, business users want to be sure the rules they've written don't have structural problems. When you have a large number of rules, checking them all by hand is not practical. Blaze Advisor includes a patented tool for verification designed for business users. This tool analyzes all the rules and other artifacts in a project to find potential problems and highlight them.

The tests identify unused variables, properties, parameters and patterns, as well as missing rules and branches (when not every value in a range or enumeration is considered). They identify problems with rules, such as rule conditions that always test true or false, rules with equivalent sets of conditions and rules that have conditions subsumed by a less specific rule. They identify potentially uninitialized variables and properties and self-contradicting test conditions, as well as infinite loops and rule firing cycles. They highlight extremely complex rule premises and various semantic errors, to help ensure best practices are being followed in the writing of the rules. Finally, this analysis is not simply done at a



surface, syntactical level—the verification will catch that “if the driver’s number of claims is less than or equal to two” is the same condition as “if the driver’s number of claims is not greater than three,” or any other way it could be logically expressed.

It is important to have automated support for verification, but it should not be a “black box.” Many errors can be found by automated verification, and the Blaze Advisor system’s verification routines are very thorough in this regard. However, in many cases automated verification can only identify *potential* problems. Effective verification must combine automation with manual consideration of potential problems. Blaze Advisor calls them “Warnings.” A review of these warnings will determine which are deliberate and necessary in a particular project and which are, in fact, problems. Your BRMS should provide a rich set of verification tools to help your business users ensure they have not made any structural mistakes.

3.3.8 Validate Business Rules

Once you have verified the rules, you need to validate that they work and do what you expect. You also want to be able to continually check that changes made to the rules as part of ongoing rule maintenance have not broken the decision. Just as you want business users to author and maintain the rules, you want the users to be able to validate the rules they are editing.

Typically this validation is divided into unit testing—checking that a change to a specific set of rules behaves correctly—and regression testing to confirm that the system as a whole behaves as expected, and that a set of changes has not broken the system.

Blaze Advisor provides a complete environment for managing tests, which is designed to allow IT and business users to effectively collaborate. The framework, known as brUnit, allows IT staff to create unit tests to detect system problems early. For example, IT staff might develop a test to ensure that deployed decision services will behave gracefully when presented with incomplete or malformed data. Using brUnit these tests can be run automatically to validate business rules before they are promoted to formal QA or production environments. Business users can not only configure and run these tests using brUnit, they can also make modifications and do all of this within their browser-based rule maintenance environment. Designing the test cases template remains a specialist task, but Blaze Advisor allows business users to configure and run them whenever they need to validate that a particular set of changes has not had unintended consequences.

Similarly, a set of regression tests can be defined in brUnit. Using the project’s standard test data, these tests are designed to confirm that the test data runs successfully through the system and produces the expected results.

3.3.9 Simulate the Business Impact

Simulating the business impact of a verified and validated set of changes is a critical requirement. After all, everything may work, but the business result may be undesirable and you need to know this before putting the change into production. FICO has a product, Decision Simulator, that does exactly that.

A typical output of Decision Simulator is shown in Figure 9. In this case, the new environ contains a set of underwriting rules that is being simulated to see how the policy volume distribution by tier will vary between the new rules and the existing old environment. A new rule is added to offer discounts to non-smokers. The simulation is analyzing the impact of this change to ensure that policy volume distribution will not be adversely skewed. In the example, a dramatic reduction in Tier B volume may not be acceptable. This kind of simulation against historical data allows a business user to understand the impact of a set of rule changes before deploying them, avoiding costly strategy errors that might otherwise be missed.

Figure 6: Results of Running a Simulation

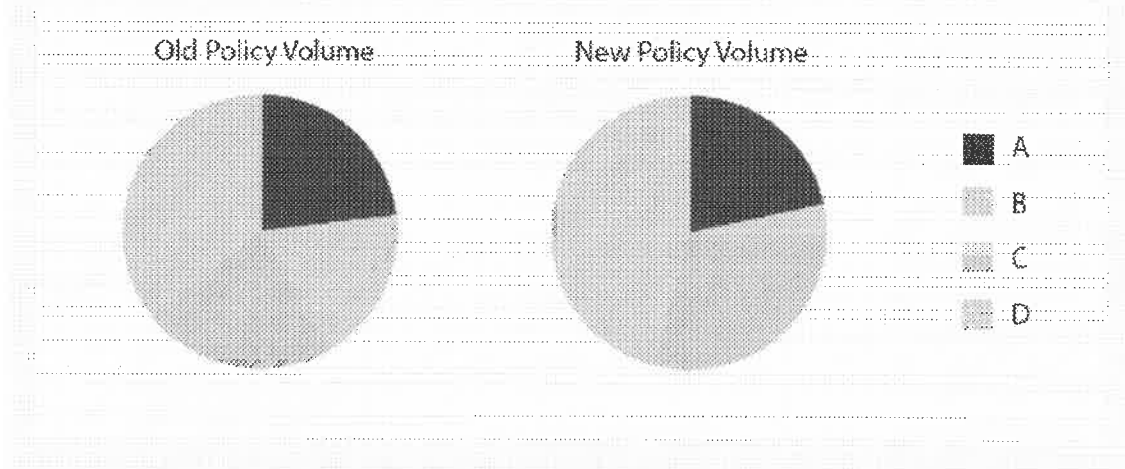


Figure 9—Results of Running a Simulation

Business users can use Decision Simulator with actual rule results to estimate outcomes and identify opportunities for improvement. They can even include simulation results within an approval process so everyone knows the impact of the proposed change in business, not technical, terms.

3.3.10 Structure for Reuse and Governance

The power of a BRMS like Blaze Advisor can grow the more it is used and as the number of rules it manages increases. But this requires that the rules are stored in a repository that is structured for reuse and governance. The repository design must support the decision service lifecycle you plan to use as well as your organization's governance policies, access controls and more. A well designed repository reduces development time and increases speed to market. It allows your decisions to be managed as a corporate asset.

FICO has developed best practices in repository design that essentially divide the repository into Technical, Business, Decision Services and Testing Libraries. These are then structured according to your business structure.

- The *Technical Library* enables reuse of infrastructure components and organizes the object model, decision service definitions, common building blocks and the overall rule architecture.
- The *Business Library* aligns rule content with business structure, grouping components by task. This enables reuse of shared business rules and the definition of specialized rules in specific areas.
- The *Decision Services Library* aligns decision services with deployment architectures so you can reuse components and package them into deployable units.
- The *Test Library* ensures that any rules or artifacts only required to test your work can be kept out of your production deployment, ensuring you deploy only what is truly required.

A repository structured this way also allows you to effectively manage your decision service lifecycle. Decision services and libraries can be packaged for release while private workspaces, supported by Blaze Advisor, can be used as sandboxes during development. The Blaze Advisor system's release



management capabilities leverage the repository design to effectively deploy into test and QA environments as well as into production to ensure that the running system stays up to date with each new release. The flexibility of the repository and the lifecycle management capabilities of Blaze Advisor ensure that you can use the software development lifecycle that works for you.

3.3.11 Operationalize Analytics and Improve Decisions

Business rules underpin your operational decisions, ensuring that decisions are made appropriately, legally and as intended. But you also want to improve your decisions, which is where predictive analytics come in. Business rules define decisions but predictive analytics make them smarter.

Historically it has been hard to integrate predictive analytics into production applications. With most predictive analytic workbenches, modelers were forced to produce a specification that would then be re-coded by hand. All the variables would have to be mapped to production data, all the calculations coded and everything would then need to be retested. This could take months, degrading the accuracy of the model, increasing costs and driving up the organization's time to market.

With Blaze Advisor you can easily integrate predictive analytics into decisions. Predictive analytic models, built using FICO™ Model Builder, can be brought into business rules-based decisions automatically using either a "black box" or a "white box" approach.

The black box approach involves the generation of code that implements the predictive analytic model and the automatic integration of this code in Blaze Advisor. Rules can then use the result of the predictive analytic model exactly as they can use other attributes or data elements. While this is a common approach for integrating models, it is not generally as effective as a white box approach as those writing the rules have no visibility into the model, the characteristics it uses, etc.

The white box approach imports predictive analytic models using a widely supported industry standard—PMML or the Predictive Model Markup Language. These imported models are available to rule developers and authorized business users who can see and even modify them using standard Blaze Advisor features. This brings the full power of the model to bear on the decision making while ensuring that those writing the rules can see and understand the workings of the predictive model.

Two of the most common kinds of predictive analytic models have specific metaphors to which they can be mapped—decision trees, discussed above, and additive scorecards. The additive scorecard metaphor—a score model—makes it easy to deploy analytic models commonly represented using this approach. As shown in Figure 10 on the next page, the score model displays different predictive attributes and their contribution to the score, which represents the likelihood of the prediction being true.

This exposes the predictive analytic model to business users, so they can view and perhaps even maintain predictive analytic models in the Blaze Advisor environment.

Each row in the scorecard represents a rule. The first one expanded, for instance, is this rule:

*If the number of claims is 0
then add 45 to the score
and add "No accidents in the last 3 years" to the reason messages*

Reason messages and reason codes are used to explain how the score was arrived at if it needs to be explained later in the decision-making process. A key difference with this metaphor is that the business user does not specify the rule based on their experience or on regulations—the specifics come from the analytic work performed as part of the model development.

Blaze Advisor makes it easy to bring the power of predictive analytics to bear on decisions, integrating judgmental and regulatory business rules with analytically derived models for more precise, more effective decision making.



FICO Recommendations Whitepaper for Chubb Business Rule COE

Figure 7: An Additive Scorecard

Characteristics	Baseline Score	Description
Gender	0	
Age	0	
Number Of Claims	0	
Base	Range	Description
0	0	45
1	1	20
2	2	0
All Other		00
Reason Code	Reason Message	
reason10	(+) No accidents in last 3 years	
reason11	(+) 1 accident in last 3 years	
reason12	(+) 2+ accidents in last 3 years	
reason13	(-) Missing accident information	
Residence Financing	0	
6-1 Years At Job	0	

Figure 10—An Additive Scorecard

3.4 Historical Challenges—Things That Add Risk to BRMS Projects

This section provides a synopsis of application areas within Personal Lines Insurance industry where BRMS has proven to be either unsuitable or unsuccessful in implementation.

3.4.1 Overview

Historically, the challenges that FICO has seen in the Personal Lines industry are from two sources:

1. Challenging Domains
2. Challenging Business Requirements

3.4.2 Challenging Domains

The good news is that the Personal Lines Insurance industry does not have a lot of application areas or domains that are particularly ill-suited for BRMS. There are lots of corporate systems (like billing) that fall outside the BRMS sphere. But specific to the business decisioning behind Personal Lines insurance, most software applications will realize benefits from BRMS.

Notable exceptions include:

1. Table-based Rating
2. Web Portal Screen Validations

3.4.2.1 Table-based Rating

The challenge with table-based rating results from 2 attributes of the approach to rating:

1. An enormous amount of data that is typically required to tier, rate, and discount policies, especially if the insurer underwrites nationally. This large amount of data needs to be managed either by a RDBMS or by a COTS product like Ratabase or Duck Creek
2. The decisioning and calculation process needs to be close to the data. It is not practical to marshal the rating data first and then call the decisioning service. The business workflow requires rules to be fired intermittent to queries throughout the rating process.



The architectural profile of Java and .NET services, especially in a SOA, makes it challenging to deploy the decisioning services close to the data. The large amount of data needed makes it prohibitive to load the tables in Blaze working memory. Hence, you are forced into an architecture that is very chatty and non-preferment.

FICO recommends using a best-of-breed COTS rating engine for table-based rating. This recommendation does not apply to score-based rating, or simple tier determination. These forms of rating are very well-suited for BRMS.

3.4.2.2 Web Portal Screen Validations

Today, just about every insurer has or is building a web portal to allow customers to apply for personal lines insurance products online. The biggest challenge to doing this effectively is to guide the customer through the application process and help him avoid making errors. This means putting the dates in the right format, selecting products that are available in his state, making sure each car has a different primary driver, etc., etc., etc.).

FICO does offer a decision management tool that lets you do screen validations, write form-fill rules, control the flow of screens/forms, etc. That tool is SmartForms. Blaze Advisor does not have the right architectural profile to do this gracefully. Certainly, the cost of the additional effort to make this work using Blaze will pay for a SmartForms license. FICO recommends using the right tool for this job.

3.4.3 Challenging Business Requirements

The biggest challenge that FICO faces in the Personal Lines Insurance industry is not related to the suitability of domains. The biggest challenges are the business requirements.

1. Build the new system exactly the way the old one was: The business wants to use a BRMS but does not permit the decision analyst to restructure the decisioning requirements so they are suitable for implementation as business rules.
2. Pre-existing requirements documentation: FICO is often brought into the project well into the design phase, when the harvesting of the business rule requirements has already been completed or is near completion. We are often forced to use client-provided business rule requirements that were not harvested in accordance with a BRMS methodology like FIRUP. Many clients failed to realize the level of flexibility in maintenance and extension that they could have if they had not forced the rules to be written the way that they specified.
3. Monolithic thinking or an inability to visualize a modular solution (a.k.a., Spaghetti Rules): FICO has many customers who do not subscribe to the notions of encapsulation and abstraction when it comes to business rule requirements. That or they define modules or components that are not appropriate for the business process model. The result is usually large complex rules, with more meta-processing than decisioning, which are difficult to maintain and extend.

There are additional scenarios. The three above are by far the most common. And the others are all just variations on the same theme:

A customer purchases BRMS technology with visions of their business realizing BRMS capabilities. They then fail to follow a proper BRMS methodology, thinking that buying the technology bought them everything. In the end, when they fail to realize the full potential of BRMS, they of course blame the technology.

The point is:

The benefits of BRMS result from the proper use of a full-featured BRMS technology like Blaze Advisor AND following a proper BRMS methodology like FIRUP throughout the SDLC (which includes the discovery and analysis of the business requirements).



3.5 BRMS Application Suitability—Guidelines for Selecting App for BRMS

This section provides guidelines for identifying application areas for BRMS within the Chubb enterprise, with special focus on Personal Lines Insurance.

3.5.1 Overview

A “business rule” is one of the least standardized concepts in the information technology industry. If you Google the term, you will find innumerable definitions, each one different from the next. FICO looks at this concept from a very broad perspective. FICO uses the term business rule only because it is part of the industry jargon. FICO really looks at business rules from the perspective of “business decisions.” So, this section will help you identify software applications with business decisioning requirements that are well-suited for Blaze Advisor.

As a rule of thumb, it is safe to assume that you are going to be successful using BRMS for automated decisioning application that would not be identified as ill suited or contraindicated under Section 3.4 above. Blaze Advisor is very broadly applicable and has a track record of success in a vast variety of application areas. Therefore, the analysis patterns described in this section are mostly intended to help substantiate a decision to use BRMS.

Content includes

1. Assessment of Volatility and Complexity of Decisioning
2. Indicators and Contraindications for BRMS
3. Sources of Knowledge
4. Types of Decisioning
5. Types of Functional Requirements

3.5.2 Assessment of Volatility and Complexity of Decisioning

There are dozens of indicators used within the business rule community to identify business domains that are suitable for the application of BRMS technologies and methodologies. However those indicators are mostly specializations of two fundamental indicators: volatility and complexity.

- **Volatility** – The relative rate at which software application requirements are subject to change within a business domain and/or the relative volume (number) of changes expected within any maintenance iteration.
- **Complexity** – The relative involvedness of the business decisioning process within a business domain. This includes the density of decision factors, the intricacy of the business logic applied to a decision, the number of exceptions to basic policies, and the relative volume of rules needed to make a decision.

The conventional wisdom is that business software applications within domains that either are highly volatile or are highly complex will benefit from BRMS technologies and methodologies. This is not to say that software applications with requirements that are not volatile and are not complex will not benefit from BRMS. Any business software application will realize some degree of benefits from BRMS. This is just saying that applications with higher volatility and/or complexity will realize greater benefits and higher ROI.

The “Magic Quadrants” graph in Figure 11 depicts the assessment process.

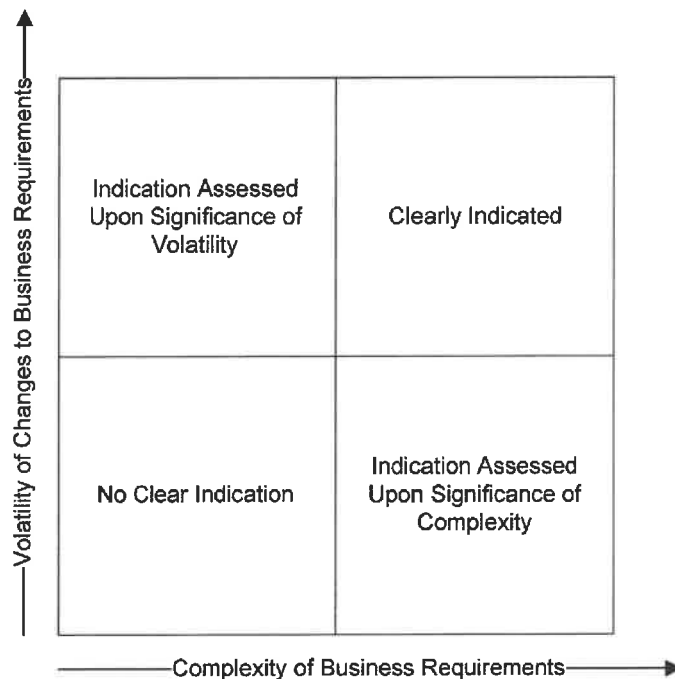


Figure 11—BRMS Indicators Based on Volatility and Complexity of Decisioning Requirements

Considering the relative complexity and volatility of business requirements in a software application's domain:

- *Upper Right Quadrant* – Applications with requirements that are highly volatile and highly complex are clear indicator for BRMS.
- *Upper Left Quadrant* – Applications with highly volatile requirements will benefit from the rule maintenance facilities offered by BRMS technologies.
- *Lower Right Quadrant* – Applications with highly complex requirements will benefit from the inference mechanisms of BRMS technologies and from the techniques for atomizing requirements that come from BRMS methodologies.
- *Lower Left Quadrant* – Applications without clear indicators of complexity or volatility may not realize as great a benefit from BRMS. Other considerations must apply:
 - Centralization of maintenance and storage using a rule maintenance application and a business rule repository
 - Sharing and reusability of rulesets
 - Encapsulation and abstraction of business logic from application logic

The above are useful indicators to assist in the evaluation of a business domain for the application of BRMS technologies and methodologies. However, considering only the combined density of volatility and complexity in the business domain is not sufficient for basing a decision. The level of difficulty expected in achieving a successful BRMS implementation and the availability of business requirements suitable for BRMS must also be considered.



3.5.3 Indicators and Contraindications for BRMS

In the next few sections, attributes of a BRMS project are described as being “Good”, “Bad”, or “Challenging”. It is important to understand the implications behind these descriptions so that they can be properly interpreted in your analysis.

- **Good** – Means that the application is well suited for BRMS. Generally, applications classified as Good for BRMS will have a significant probability of realizing benefits from BRMS.
- **Bad** – Means that the application is poorly suited for BRMS. Generally, applications classified as Bad for BRMS will have a significant risk of failing to realizing benefits from BRMS.
- **Challenging** – Means that the application may in fact be well suited for BRMS, but something about the attribute induces complications (challenges) in implementation. Generally, applications classified as Challenging for BRMS will be successful in implementing BRMS. They are considered challenging because of the challenges the team will face.

Generally, the types of complications that we are trying to avoid are the following:

- **RMA Complexity**—if the resulting RMA is going to be too complex, BRMS may be contraindicated. Complex RMAs can be very expensive to build and/or maintain, are of little value to the business users, and often encounter challenges during product version upgrades (resulting from coding to the low-level API)
- **Performance Problems**—rules are generally fast enough; but some domains require faster speed and other domains have rule complexities that impact performance. You do not want to choose BRMS for domains that need to mull-over huge numbers of database queries. You also do not want to implement complex aggregations of data.
- **Unmanageably High Rule Volumes**—no matter how powerful your BRMS technology is, no one can effectively maintain 300,000 rules. We try to avoid application area that require unmanageably high rule volumes and leave those domains to the RDBMS tools.
- **Need for Extensive Knowledge Engineering**—if the problem domain requires abstract reasoning, then Knowledge Engineering is needed. KE is software engineering discipline that is to Decision Requirements Analysis like brain surgery is to healthcare. We prefer not to do this kind of analysis for large scale business applications. It is time consuming and not practical for a large complicated set of decisions.

The following sections look at these indications and contraindications for a number of decision factors in determining the suitability of BRMS for a project.

3.5.4 Sources of Knowledge

The first and most important decision factor in deciding whether BRMS is suitable for an application is the source for the knowledge that will be implemented in the decisioning.

Table 5—Sources of Knowledge for BRMS

Good	Bad	Challenging
<ul style="list-style-type: none"> • SME or Domain Experts • Policy Manuals • Regulations • Regimens • Traditions 	<ul style="list-style-type: none"> • Databases • Guidelines • Opinions 	<ul style="list-style-type: none"> • Source Code • Specialists • Novices • End-users of existing applications • Technicians



3.5.4.1 Good Sources of Knowledge

The best sources for knowledge are Subject-matter Experts and Domain Experts. Published (internal or public) regulatory or governance materials are also excellent sources for policy-based knowledge. Documented regimens and/or description of traditions are good sources for practice-based knowledge provided the source for the explanation of traditions is credible.

Regimens and traditions are becoming more and more common. Today's SME are trending more toward being experienced users of existing systems. This new breed of SME knows how things are done. They know what works and what does not work. They just don't have the deep knowledge behind why things are done the way they are done. If the process is not broken, and the project is to a new system, then regimens and traditions are perfectly acceptable sources of knowledge for BRMS.

3.5.4.2 Bad Sources of Knowledge

Throughout the 1980's and 1990's, it was a common practice to break decisioning processes down into database lookups. This made sense at the time because SOA was not yet conceived and it allowed IT to effectively separate business decisioning from application code. It also allowed the business to maintain and extend the decisioning by editing database records. It was actually a form of early business rule management.

These systems are now being modernized and BRMS is being considered as a replacement for the database. That is fine, so long as the system is suitable for BRMS. The bad comes in when the decision is made to use the legacy database as the source of knowledge for the business rules. The knowledge that was used to create the database lookups is so buried in the implementation that there is really no practical method for pulling it out.

Another bad source of knowledge can often appear masked as a good source. Be careful that published regulatory, governance, and regimental materials are in fact documentation of policy-based knowledge and not published guidelines. Guideline rules typically have orders of magnitude more exception-handling rules than guideline rules.

Also, be careful of "opinions" during the discovery and analysis of traditions; opinions are not reliable sources for practice-based knowledge. Many people wonder why medical systems do not use BRMS to assist in diagnosis. The technology seems to be a perfect fit and system would have high value and benefit. The reason is that doctors are the worst source of knowledge for precisely this reason.

3.5.4.3 Challenging Sources of Knowledge

Some challenging sources of knowledge are human sources that are not true subject-matter experts or domain experts.

Specialists – The challenge with using specialists is that you must use many specialists to acquire all the knowledge for a domain. Few specialists are broad enough to understand the interaction of various aspects of a domain. Therefore, it is challenging to spot commonalities or opportunities for sharing and reuse. It is also common to get conflicting knowledge in areas where two specialties overlap. This is because each specialist views the overlapping area from the perspective of their specialty.

Novices – There is nothing inherently wrong about a novice SME. Every Domain Expert was once a novice. However, they do present challenges as sources for domain knowledge and slow down the discovery process with delays for verifications. Project planners must account for such challenges and delays or this could affect the project timeline and delivery schedule.

End-Users – End-users of existing software applications are not subject-matter or domain experts. They are simply highly experienced in using the software application that the BRMS application is replacing. End-users can generally provide effective explanations about "what" the legacy system does, or how it works. However, end-users rarely understand the business policies, practices, and procedures that underlie the decisioning within the legacy system. Therefore, unless you are building a point-for-point replacement of a legacy system, they offer little value as subject-matter experts.



Technicians – The challenge with using technicians for subject-matter expertise is their tendency to try to do the business rule analysts job for them. Technicians often dive down from the conception or policy-based level and incorporate their recommendations for implementation in with their subject-matter expertise. This behavior often prejudices inexperienced business rules analysts toward inappropriate implementations. It takes an experienced business rule analyst to explain tactfully why they need to reverse engineer the 40,000-row decision table that the technical SME spent two weeks putting together for them.

Legacy Source Code – This used to be considered a bad source of knowledge. The notion of mining business rules from any legacy implementation was a notoriously bad idea. This is because the decision analyst community did not regard legacy source code or legacy database triggers and stored procedures as effective sources for knowledge about a business domain. In recent years, this source has moved from bad to very challenging. Business analysts have learned techniques for mining rules and tools for assisting in the process are becoming mature. It is now possible to deliver successful BRMS applications with decision requirements that were mined from legacy source code.

However, to be successful, expectations must be managed. You cannot simply grab all the “if-then” logic out of COBOL and dump it in Blaze and expect to have an effective BRMS. The specification of business requirements in the legacy code was probably done years before the first notions of business rule analysis came into practice. It is not appropriate to assume it is ready to go as-is for implementation in BRMS. Rule mining is a long process that feeds business rule analysis. Rule mining does not replace decision analysis.

3.5.5 Types of Decisioning

The second most important decision factor in determining the suitability of BRMS for an application is the type of decisioning that will be performed by the application.

Table 6—Types of Decisioning for BRMS

Good	Bad	Challenging
<ul style="list-style-type: none"> Analytical Qualification Classification Exception-handling Pattern Recognition 	<ul style="list-style-type: none"> Algorithmic Iterative Recursive Numerical Statistical 	<ul style="list-style-type: none"> Circumstantial Quantification Certainty Analysis Scoring

3.5.5.1 Good Types of Decisioning

The best type of decisioning for BRMS is simple cognitive logic. The following are some ideal examples of decisioning that meet this criterion:

Simple Analytical Reasoning – Simple chains of atomic rules the work together to reach a conclusion. i.e.,

- If the available balance of the account is less than \$0, then the account is overdrawn. If the account is overdrawn, then charge a NSF fee. If the account is overdrawn, and the pending transactions balance is greater than \$0, and the customer is a VIP, then do not charge a NSF fee.

Qualification – Sets of (sometimes numerous) rules that work collectively to determination a Boolean fact about a business concept. i.e., A set of rules to determine if an exchange transaction is or does not comply with SEC suitability regulations.



Classification – Sets of rules that specify the conditions for classifying a business concept as a member of a specialized subset. i.e.,

- If the total balance of all household accounts for a customer is greater than \$50,000, then the customer is a VIP.

Exception Handling – Sets of rules that define exceptions (special circumstances) to basic policies. i.e.,

- The maximum debt-to-income ratio for a HUD-conforming mortgage is 2.75. If the effective loan-to-value ratio is less than 0.5, then the maximum debt-to-income ratio is 3.25.

Pattern Recognition – Sets of rules that specify the condition by which to recognize patterns. These rules often identify circumstances where certain business actions are necessitated. i.e.,

- If the credit card balance is over limit, and the account history show at least three over-the-limit events in the trailing 12-month period, and the 180-day derivative of debt to income is inclining (greater than zero), then poor debt management is suspected. Recommend credit counselor contact for debt management.

3.5.5.2 Bad Types of Decisioning

Business rules management technologies, especially Rete-based, are not well suited to programmatic logic. This is because Rete engines break rule logic down into atomic components within the rete network and then apply an inference algorithm to that network. However, some types of business knowledge do not break down gracefully into an atomic structure. These include:

Algorithmic Logic – Formulae for calculating business quantities from business data, i.e., the amortized value of an annuity.

Iterative Logic – Formulae of applying logical test and/or operations to a collection of business concepts. The actual test may be suitable for implementation as business rules. However, the process of looping over a collection (i.e., batch processing) is not suitable for implementation within the business rules and should be implemented external to the rule service.

Recursive Logic – Formulae for recursive operations such as sorting are best implemented within application code.

Numerical Methods – Business rules may use numerical quantities (i.e., such as derivatives, integrals, etc.) in their determinations. However, the calculation of those quantities is best implemented within application code.

Statistical Methods – Business rules may use statistical quantities (i.e., such as mode, mean, etc.) in their determinations. However, the calculation of those quantities is best implemented within application code.

“Advisory” systems where the approach to giving advice is not clearly defined. FICO faced challenges at Chubb in the UK where the rules were just a huge cloud of advice strings and no real decisioning going on.

Conducting Q&A dialogues can be challenging, in spite of the “expert systems” history of BRMS, because it either requires the BRMS to keep the state of the dialogue or needs the entire conversation to be passed in with each call. And BPM, for much the same reason.

Any application which needs extensive use of databases conditional on decisioning: e.g. large banks of reference data or looking for patterns across multiple entities (e.g. fraud). This can lead to serious performance issues and needs clever technical solutions.

3.5.5.3 Challenging Types of Decisioning

Most challenging types of decisioning are complex cognitive processes. While perfectly appropriate for implementation using a rete-based BRMS, complex cognitive logic is challenging. Generally, the business community will not be able to maintain these rules. Moreover, the IT community will likely



require assistance for an experienced Knowledge Engineer to help them implement these types of solutions.

On the other hand, BRMS do offer an effective solution to these problems. Many software applications (i.e., automated insurance underwriting) were not achievable before the advent of BRMS. Therefore, Fair Isaac does not recommend you to discount such application area as candidates for BRMS. Fair Isaac simply recommends that you seek professional Knowledge Engineering support for the following types of decisioning processes:

Circumstantial Evidence – Decision processes based upon weighing contributing facts as a body of circumstantial evidence.

Quantification – A qualification and/or classification decision that involves a statistical and/or numerical assessment, i.e.,

- If the probability of foreclosure is HIGH, then defer the book of business.

Certainty Analysis – Similar to circumstantial evidence, certainty analysis considers evidence indicating a determination against evidence contraindicating a determination. The result is a confidence factor representing the overall certainty in the determination. This is Boolean with “gray areas.”

Scoring – you should only implement scoring models using the Score Card metaphor included in the Blaze Advisor product. You should only approach the implementation of a custom scoring algorithm under the guidance of an experienced Knowledge Engineering professional with a core competency in score modeling.

3.5.6 Types of Functional Requirements

The final decision factor in determining the suitability of BRMS for a software application is the types of functional requirements.

Most business analysts are familiar with the difference between functional and non-functional requirements. There is a natural tendency to try to put all of the functional requirements into the BRMS. That makes sense only on first glance. In practice, there are only certain types of functional requirements that are suitable for implementation in the BRMS.

Table 7—Types of Functional Requirements for BRMS

Good	Bad	Challenging
<ul style="list-style-type: none"> • Classification • Boolean Determination • Diagnosis • Prescription • Configuration Mgmt • Condition/Response 	<ul style="list-style-type: none"> • GUI Format/Control • Entity Translation • Data Mining or Aggregation • Workflow • Process Control (beyond ruleflow) • Batch Loop 	<ul style="list-style-type: none"> • Validation • Verification • Cleansing • Calculation • Lookups

3.5.6.1 Good Types of Functional Requirements

The good news is; the good types of Functional Requirements are very broadly applicable to most any business software application.

- **Classification**—decision services that classify one or more business concepts as being a “type of” something for the purpose of prescribing a treatment that is dependent upon that type. i.e., tiering for rate calculation.



- **Boolean Determination**—decision services that make a Boolean (yes/no, pass/fail, go/nogo, good/bad) decision that will drive the disposition of a business object.
i.e., regulatory compliance.
- **Diagnosis**—decision services that analyze attributes and facts about a business object and to make a determination about the state of that object
i.e., fraud detection.
- **Prescription**—decision services (typically fed by classification and diagnostic services) that prescribe the best treatment or disposition for a business object
i.e., risk-based underwriting.
- **Configuration Management**—decision services that whether and/or how two or more business objects will either interoperate or be configured relative to one another
i.e., endorsements and discounts.
- **Condition/Response**—a very simple form of a prescription service where the response is indicated by the recognition of a simple pattern in the attributes of a business object with very little to no analysis
i.e., knock-out rules

3.5.6.2 Bad Types of Functional Requirements

The good news is; the bad types of Functional Requirements are requirements in most every business software application. It is a virtual certainty that you will not be able to put all of your functional requirements into the BRMS.

- **GUI Format/Control**—rules about format of dates and money, or required fields, or filtering lists based upon values in other fields. Recommend using SmartForms.
- **Entity Translation**—rules about simplifying and/or flattening a complex data object model such that pertinent information is readily available. Recommend externalizing from rule services and using a best-of-breed commercial off-the-shelf (COTS) tool.
- **Data Mining or Aggregation**—rules about assembling a coherent data object from two or more disparate data sources. Recommend externalizing from rule services and using a best-of-breed COTS tool.
- **Workflow**—rules about the sequence of events within a business software application. Recommend externalizing from rule services and using a best-of-breed COTS tool.
- **Process Control (beyond ruleflow)**—rules about how to process or dispose of events within a business software application. Recommend externalizing from rule services and using a best-of-breed COTS tool.
- **Batch Loop**—conditions under which a business object is selected for participation in a batch process. These selection criteria are typically simple enough to code directly stored procedure.

3.5.6.3 Challenging Types of Functional Requirements

Organizations with a strong COBOL heritage typically face challenges implementing business rule services because they are not used to separating the management of data from the decisioning on data. It is very common to see an approach where data is obtained, all the decisions that pertain to that data are made, and then the next piece of data is obtained, and so on. As a result, it is very common to see data validation, verification, and cleansing logic embedded with true business decisioning. In BRMS, we call these spaghetti rules.

Validation, Verification, and Cleansing services are actually very simple and straight forward to implement if broken apart into encapsulated and abstracted rulesets. They become very challenging when muddled together.



Calculation services can be rule-based, look at TurboTax. These are very challenging systems to build. It takes a lot of experience in decision analysis and business rule architecture to properly encapsulate and abstract the various types of decisioning in complex calculation services. FICO strongly recommends the employment of highly experience analysts and architects on such services. When designed and implemented properly, many Calculation services have achieved great success and received a lot of value from BRMS.

Lookup services are on the borderline between Bad and extremely challenging. Most Lookup services should be implemented in the database. FICO does recognize a few circumstances when lookups should be done within the BRMS services. All 3 of the following must be true, otherwise it is not recommended:

1. The volume of reference data is relatively small, no more than a few 100 Mbytes.
2. The reference data is only meaningful to the BRMS application and no other applications will ever look at or change the reference data.
3. There are significant benefits to a single and centralized point of persistence and maintenance.

If all of the above are true, then it is OK to implement the lookup within a Blaze decision services. However, it is recommended that you consult FICO for best practices on implementing lookups. Improperly implemented lookups (i.e., as decision tables) can have detrimental impacts on performance, maintainability, etc.



4 Overview of FICO's Decision Harvesting Methodology

This section provides an overview of FICO's FIRUP Methodology as it applies to the analysis and definition of decisioning requirements. This overview includes the following subject-matter:

1. FICO's FIRUP Delivery Methodology
2. FICO's Decision Harvesting Methodology
3. Decision Requirements Analysis
4. Decision Requirements Analysis Workshop
5. Decision Requirements Diagram (DRD)
6. Decision Definition Document (DDD)
7. Supporting Requirements Document
8. Decision Harvesting (a.k.a., Rule Harvesting)
9. Decision Harvesting Workbooks
10. Ancillary Rule Harvesting Activities and Artifacts

4.1 Overview

The purpose of Decision Harvesting is to identify and capture the decisioning requirements for the given project or iteration. These requirements form the basis of communication and agreement between the stakeholders and the project team regarding the project's or iteration's deliverables.

Requirements gathering is a fairly broad topic and the detailed processes will vary significantly based upon the type of solution being considered. The FIRUP Methodology is therefore intentionally written to provide a very high-level set of expectations around requirements gathering.

Early in the project, as part of planning sessions, it may be necessary to define high-level requirements in such a manner as to identify the functional increments and architectural components relevant to the overall project. These functional increments would be assigned to specific iterations. Within each iteration, the assigned functional increment would go through further requirements definition to expand on the details.

FICO's decision harvesting methodology is designed to

- Document specifications for decisioning requirements in a manner appropriate structured for implementation within Blaze Advisor
- Provide traceability to ensure that all decision-based requirements are met
- Maximize the maintainability and extensibility of the business rules.

FICO's methodology is highly iterative, JAD-like, and involves a frequent ongoing interactions with business subject-matter experts throughout the project's inception and elaboration phases (sometimes, when requirements are complex or extensive, well into the construction phase).



4.2 FICO's FIRUP Delivery Methodology

It is important to begin with an overview of certain aspects of FICO's delivery methodology because many aspects of FICO's methodology for performing decision requirements analysis are strongly coupled to the FIRUP approach. The subsections in this overview will elaborate on this coupling between project activities, disciplines, and project approach.

FICO's Project Delivery Methodology (FIRUP) is based on the Unified Process, which is a widely recognized industry standard delivery methodology in use by an increasing number of FICO's clients and partners. In keeping with its open doctrine, FICO has adapted the Unified Process to meet the unique needs of the FICO product suite: leveraging proven best practices from the Unified Process, the Project Management Institute's Project Management Body of Knowledge (PMBOK) framework, and from more than 50 years of experience in implementing decisioning and analytic solutions throughout the world.

4.2.1 Iterative Project Management

The FIRUP methodology is equally applicable to both traditional cascade (waterfall), incremental, and iterative project management philosophies.

1. *Traditional (waterfall) SDLC* – The waterfall model is a sequential process in which progress is seen as flowing steadily downwards (like a waterfall) through the phases of Conception, Initiation, Analysis, Design, Construction, Testing and Maintenance.
2. *Incremental (Agile) SDLC* – Incremental models (like Agile) are based on the concept of allowing an implementation to evolve over a series of rapid incremental development sprints through collaboration between self-organizing, cross-functional teams.
3. *Iterative (RUP) SDLC* – Iterative models (like RUP) are based on the concept of delivering an implementation through a series of lifecycle phases associated with the maturity of capabilities, business value, and risk mitigation.

While FIRUP does not require iterations, FIRUP provides the flexibility to leverage iterations where a portion of the solution is constructed and tested to specifically address significant risks and to deliver increasing business value throughout the SDLC.

FICO's experience over the past 50 years has proved the iterative approach to be the best approach for delivering decisioning solutions.

4.2.1.1 FIRUP Lifecycle Phases and Iterations

The FIRUP methodology employs four distinct phases to project lifecycle, as shown in the below illustration. Each phase focuses on a key aspect of delivering a successful solution utilizing a variety of disciplines (described in the next subsection).

Within each phase, a FIRUP project may be divided into fixed spans of time called "iterations." Each iteration ends with a new, incremental delivery. Very early in the project, during the inception phase, a new delivery may only consist of documentation or a proof-of-concept prototype, but as the project progresses, the focus shifts to incremental deliveries of a working solution that adds capabilities to what was delivered in the previous delivery.

By delivering iteratively and incrementally, the FIRUP methodology ensures that the risks are being surfaced and confronted as early as possible, and that we stay focused on our principal of delivering business value to the business as early and as often as possible. As this figure illustrates, risk drives the iterations.

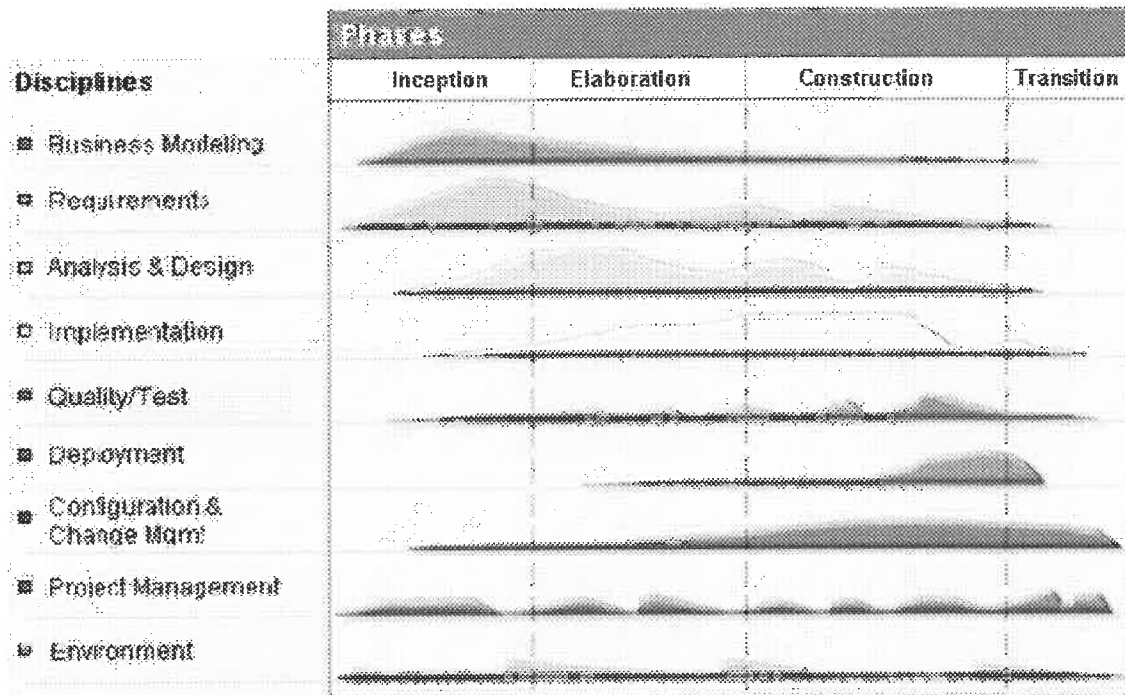


Figure 12—FIRUP™ Project Lifecycle Phases

The benefits of incremental delivery through the use of iterations include:

- Business users are involved earlier and see functionality being built incrementally, reducing late-project surprises
- The higher-valued functionality is delivered early, reducing the impact of later changes in business priorities and potential budget cuts
- Risks are mitigated earlier in the project, reducing late-project surprises

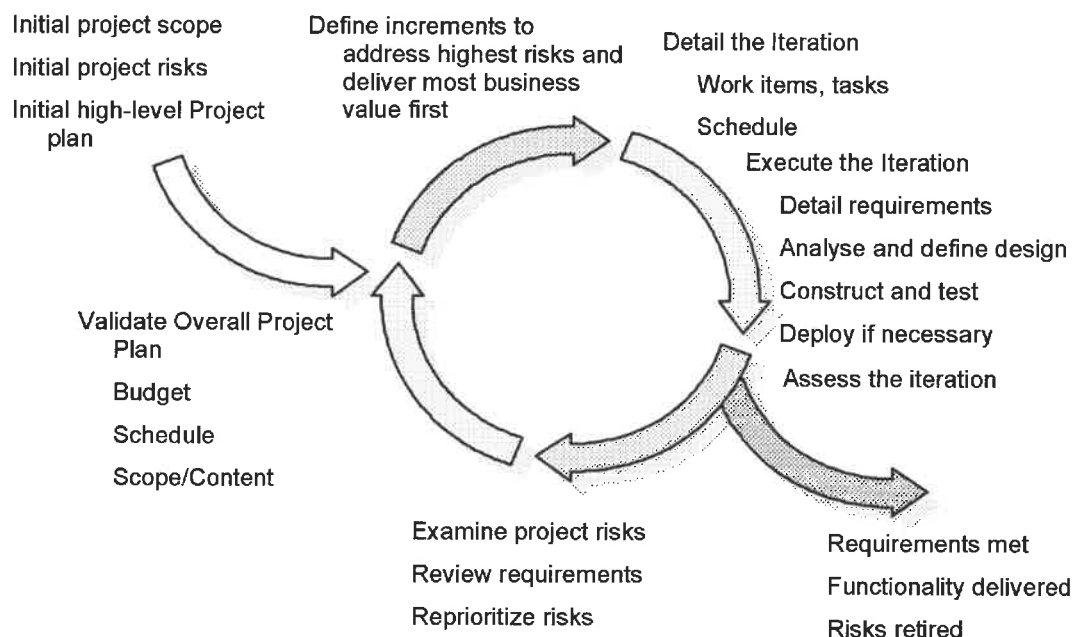


Figure 13—FIRUP™ Project Iterations

4.2.2 FIRUP Disciplines

The left column in Figure 12 (above) list nine disciplines associated with the development of a BRMS. These disciplines provide processes that underpin the methodology providing a comprehensive set of documented practices, tools, and techniques to ensure timely delivery in an efficient, cost effective, quality-driven manner. This section discusses each and how the disciplines are interrelated.

A Discipline is a collection of Tasks that are related to a major "area of concern" within the overall project.

This approach aligns Tasks to a particular competency area and group of project team members. Although it is more common to perform Tasks concurrently across several Disciplines (for example, certain requirements Tasks are performed in close coordination with analysis and design Tasks), separating these Tasks into distinct Disciplines is simply an effective way to organize content, which makes comprehension easier.

- *Project Management* – addresses project planning, risk and issue management, communications, change management, and other areas of project governance
- *Business Modeling* – addresses the business side of things. Includes such items as: business strategy, organizational structure, roles and responsibilities, operations, business processes, business metrics and performance improvement
- *Requirements* - documents the processes to elicit, analyze, specify, validate, and manage the requirements for the solution to be developed
- *Analysis and Design* – defines and documents how the FICO solution is designed to meet each business requirement and how it fits into the technical system architecture



- *Implementation* – defines the steps to build and configure the solution and verify that the components function as specified
- *Quality Management and Testing* – defines the processes that assure the solution meets the defined requirements
- *Deployment* – provides appropriate tasks to deploy the solution into the test and/or development environment(s)
- *Configuration & Change Management* – controls the changes to key project artifacts and components of the solution, ensuring their synchronized evolution
- *Environment* – specifies and configures the physical machines and logical environments in which the application will be developed, tested, and deployed into production

It is important to understand that all of the above are disciplines/competencies that are expected of all team members. They are not roles that individuals fulfill, but rather, are attributes of a proficient team member.

E.g., the project team will certainly have one or more team members functioning as project manager and team leaders. These individuals would be expected to have core competencies in the Project Management discipline. However, as proficient team members, those same individuals are expected to demonstrate capabilities in requirements, analysis, design, implementation, etc.

4.2.3 Key Disciplines for Decision Requirements Analysis

Again, all disciplines collaborate in the performance of activities on a project. However, there are some disciplines that are more instrumental to the performance of certain activities than others. For Decision Requirements Analysis, the key disciplines are:

- Business Modeling
- Requirements
- Analysis and Design

Of course, Project Management is always a critical component of all activities. And, the purpose of requirements analysis is to feed the implementation activities. And, part of requirements analysis is the specification of testing/validation requirements for Q/A. And so on. For the sake of conciseness, this discussion will assume those dependencies as obvious and aligned with traditional projects. The remainder of this section will focus on the three disciplines identified above.

4.2.3.1 Business Modeling

The purposes of business modeling in the FICO Unified Process are:

- To understand current problems in the target organization and identify improvement potentials that may be served by BRMS technologies and/or methodologies.
- To understand through modeling the current and desired business processes, data, and rules of the target organization to aide in developing requirements for BRMS technologies and/or methodologies.
- To ensure that business, users, developers, and other parties have a common understanding of the organization.

The work breakdown structure for business modeling is show in the figure below. There are 5 major activities associated with this discipline:

- Define Business Goals
- Model Business Organization

- Model Business Processes
- Model Business Objects
- Model Business Object States

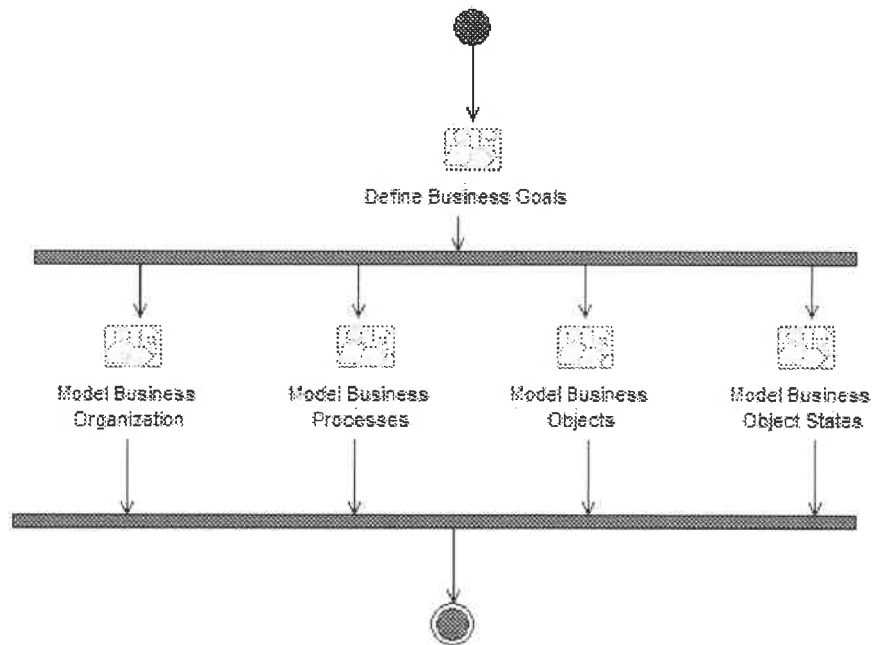


Figure 14—WBS for the Business Modeling Discipline

To build a proper BRMS solution, the decision requirements analysis team needs to understand the business's organization, including their problems, goals, processes, business objects, data, and rules. The process of formalizing this understanding is called Business Modeling.

- Determine requirements approach
 - Formality, traceability, etc.
- Analyze the problem
 - Stakeholder requests
 - User research
 - Business process modeling
 - Industry analysis, etc.

A business model should reveal both the static structures of the target organization, but also the dynamic behavior of that organization. The static view of the organization models the things that comprise the organization and the information, relationships, and rules about those things. The static view of the organization can involve its organizational units, such as divisions, departments, and locations, but also its business objects, such as customers, accounts, payments, and products. UML class diagrams are often used to model these elements.



The dynamic view of a business models how the business behaves. The dynamic view is revealed in the businesses processes which describe how the business operates, but also in state models of key business objects. UML activity diagrams can be used to model business processes, while UML state machine diagrams model business object states, their transitions, rules, and actions.

4.2.3.1.1 *Relation to Other Disciplines*

The business-modeling discipline is related to other disciplines, as follows:

- The Requirements discipline uses business models as an important input to understanding the requirements of the system.
- The Analysis & Design discipline uses business models as an input to defining software systems that seamlessly fit into the organization.
- The Deployment discipline uses business models as an aid in planning the deployment of a software system.
- The Environment discipline develops and maintains supporting artifacts, such as the Business Modeling Guidelines.

4.2.3.2 **Requirements**

The purpose of this discipline is to:

- Understand the problem to be solved
- Understand stakeholder requests (what stakeholders want)
- Understand stakeholder needs (needs are indeed different than requests)
- Define Project Vision
- Define the requirements for the solution (what the system must do)
- Define the boundaries (scope) of the system
- Identify external interfaces for the system
- Identify technical constraints on the solution
- Provide the basis for planning iterations
- Provide the initial basis for estimating cost and schedule
- Develop Requirements Management Plan to manage system scope
- Model Usage—define both actors and usage (actor roles)
- Produce detailed functional requirements (including use cases)
- Define non-functional requirements and begin specifications in the Supporting Requirements Document (SRD)
- Model the functional flow of the system
- Define information architecture (screen-flow diagrams and wireframes)
- Review requirements with stakeholders and get buy-in/sign-off

The work breakdown structure for the Requirements discipline is show in the figure below.

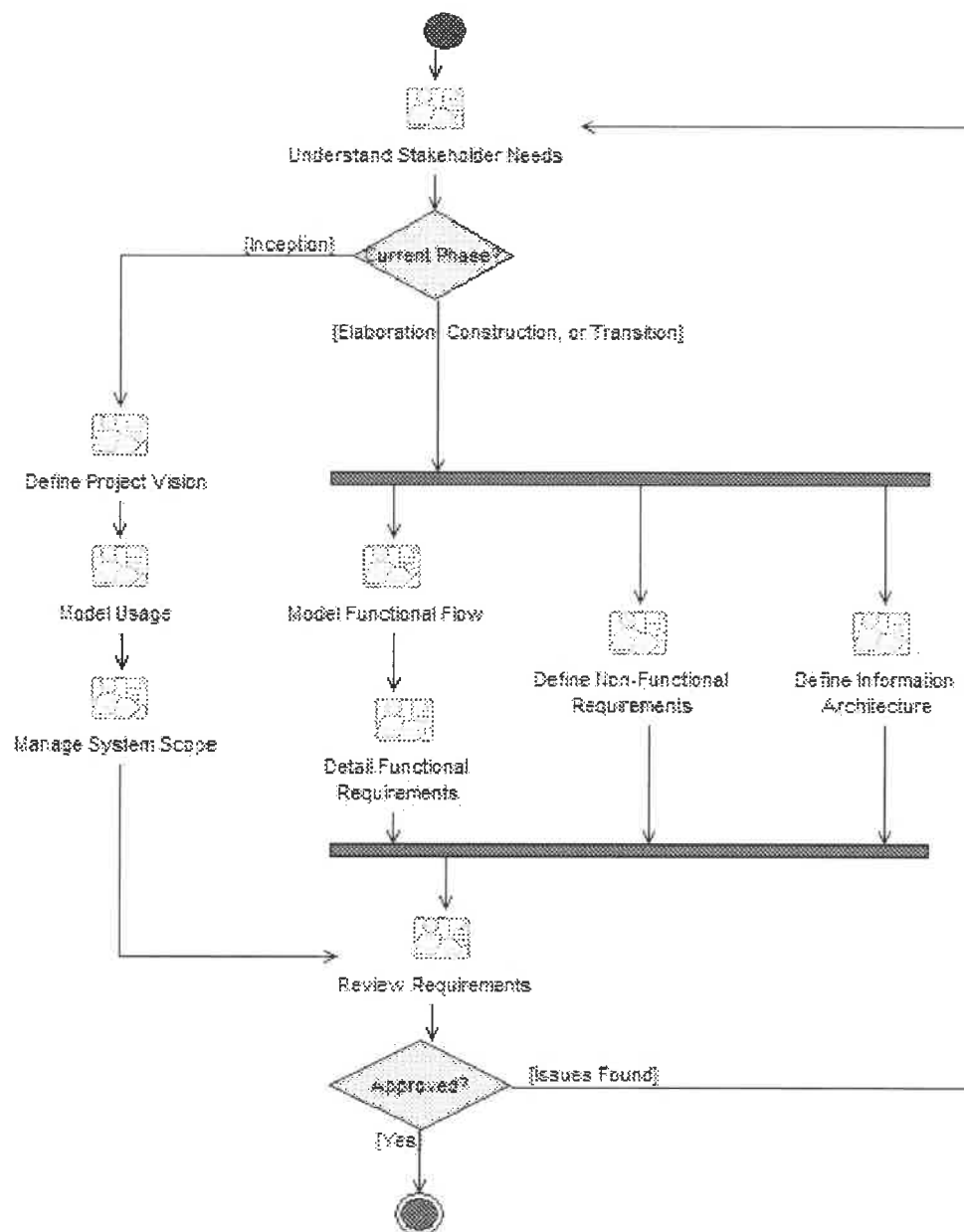


Figure 15—WBS for the Requirements Discipline

To achieve these goals, it is important to understand the definition and scope of the problem that the business is trying to solve. Identify Stakeholders and define the problem to be solved.

Having agreed on the problem to be solved, the Requirements for the system are elicited, organized, analyzed, validated, and specified.



Throughout the lifecycle, the team must manage changes to the requirements.

4.2.3.2.1 *Relation to Other Disciplines*

The Requirements discipline is related to the other disciplines in the following ways:

- The Analysis and Design and Implementation disciplines get their primary input from the Requirements discipline.
- The Test discipline validates the system against the requirements.
- The Configuration and Change Management discipline provides the mechanisms to manage changes to the requirements.
- The Project Management discipline plans the project and assigns requirements to each iteration by analyzing the prioritized requirements and assigning work.

4.2.3.3 **Analysis and Design**

This discipline is concerned with establishing a feasible vision for the system and assessing appropriate techniques for building the solution. This involves transforming Requirements discipline work products into the work products specifying the design of the software the project will develop.

The goals of Analysis & Design are:

1. To transform the requirements into a design of the system-to-be.
2. To evolve a robust architecture for the system.
3. To adapt the design to match the implementation environment, designing it for performance.

The main activities performed within this discipline are:

- Define Candidate Architecture
- Construct, Execute, and Evaluate Proof-of-Concept (POC)
- Review and Approve Candidate Architecture
- Iteratively Refine Architecture
- Iteratively Analyze System Behavior (both functional and non-functional)
- Iteratively Analyze User Experience (if applicable) and Systems Impact
- Continually Update Architecture
- Design System Components
- Design Services (following a Services-Oriented Philosophy)
- Design Data Sources and Back-end Services
- Design User Interfaces (e.g. Rule Maintenance Application) and System Integration Points

All of the above are very broad activities that incorporate a myriad of best practices and produce numerous deliverables. Drilling down into each is beyond the scope of this document.

It is important to note that the analysis of user experience and design of user interfaces also includes the rule maintenance application (RMA) component of the BRMS.

The work breakdown structure for the Analysis and Design discipline is show in the figure below.

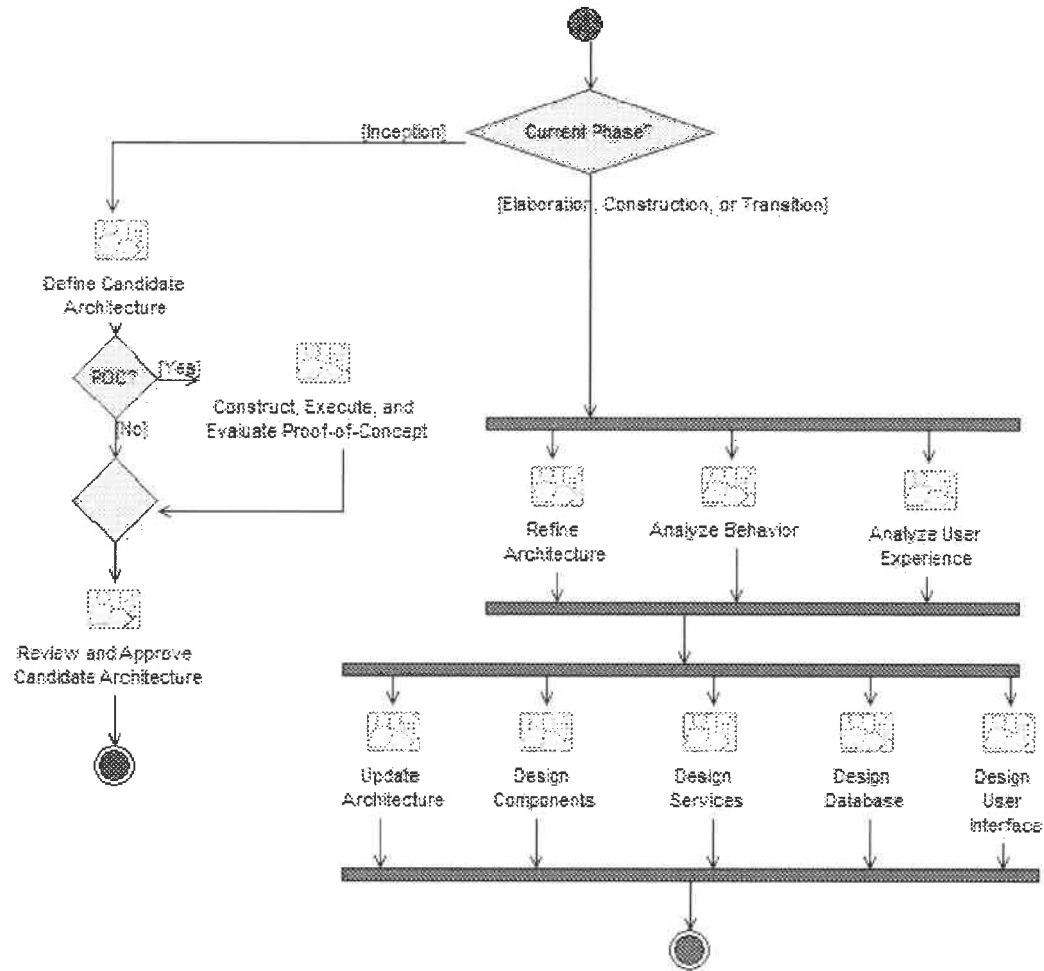


Figure 16—WBS for the Analysis and Design Disciplines

As the name implies, Analysis & Design is composed of three related activities: analysis, architecture, and design.

4.2.3.3.1 Overview of Analysis

Analysis is the examination of the problem to reveal the underlying

- Abstractions,
- Data,
- Behaviors,
- Relationships,
- States,
- Rules, and
- Constraints.

Analysis is not the definition of a solution, but rather the understanding of a problem. Thus, analysis does not discuss implementation technology.



4.2.3.3.2 Overview of Software Architecture

Software Architecture is the set of significant decisions about the structural organization of a software system

- Compositions of Services and Components
- Global Control Structures
- Protocols for Communication, Synchronization, and Data Access
- Assignment of Functionality to Design Elements
- Physical Distribution of System Elements
- Selection of Design Alternatives, Preferred Mechanisms, and Style

Software Architecture balances these competing objectives:

- | | | |
|-------------------|--------------------------------------|--|
| • Usage | • Commonality | • Technology tradeoffs and constraints |
| • Functionality | • Reuse | • Aesthetic concerns |
| • Performance | • Extensibility | • Operability |
| • Maintainability | • Comprehensibility | |
| • Reliability | • Economic tradeoffs and constraints | |
| • Scalability | | |
| • Resilience | | |

4.2.3.3.3 Overview of Design

Design merges the domain models, business rules, and constraints revealed by analysis with the strategic technical decisions made by architecture to model an implementable solution to a set of requirements and quality constraints at a logical level above code. Software design focuses on many issues

- | | |
|--|-------------|
| • User interface, | • security, |
| • persistence, | • logging, |
| • object distribution and communication, | • etc. |
| • legacy systems integration, | |

Architecture will have made many decisions about these issues, but it is design that brings those decisions to bear on the application

4.2.3.3.4 Relation to Other Disciplines

The Analysis & Design discipline is related to other disciplines, as follows:

- The Requirements discipline provides the primary input for Analysis & Design.
- The Implementation discipline implements the design.
- The Test discipline tests system designed during Analysis & Design.
- The Environment discipline develops the modeling standards and organization and tool guidelines and chooses and administers the tools that are used during Analysis & Design.
- The Project Management discipline plans the project, and each iteration (described in an Iteration Plan).



4.3 FICO's Decision Harvesting Methodology

Decision harvesting is a fairly broad topic and the detailed processes will vary significantly based upon the type of solution being considered. Guidance for this activity is therefore intentionally written to provide a very high-level set of expectations around decision harvesting.

FIRUP defines a requirement as:

"A requirement is formal statement of:

- an attribute to be possessed by the product or a function to be performed by the product.
- the performance standard for the attribute or function.
- the measuring process to be used in verifying that the standard has been met."

Informally, requirements are the project team's to-do list.

Requirements define what is needed and focus the project team. They are the primary method used to communicate the goals of the project to everyone on the team.

Requirements define:

- What the users want
- What the system must include to satisfy the user and business needs
- How one will demonstrate that the requirements have been satisfied

Requirements are the basis for capturing and communicating needs, managing expectations, prioritizing and assigning work, verifying and validating the system (acceptance), and managing the scope of the project.

Requirements may take different forms, including Use Cases and Scenarios, unstructured text, structured text, or a combination, and they may be stated at different levels of granularity. At the highest level of abstraction, Features define the services that the system must provide to solve the business problem.

A system must perform according to the behavior that requirements specify. However, there are system requirements that do not represent a specific behavior:

- Legal and regulatory requirements, as well as application standards
- Quality attributes of the system to be built, including usability, reliability, performance, and supportability requirements
- Interface requirements to be able to communicate with external systems
- Design constraints, such as those for operating systems and environments and for compatibility with other software

These quality requirements are often referred to as nonfunctional requirements.

Quality requirements that apply to the system as a whole are captured as structured text in Artifact: Supporting Requirements. Quality requirements that are closely associated with a particular Use Case are often captured in the Use Case itself to simplify review, understanding, and maintenance.

4.3.1 Decision Requirements Harvesting Techniques

This guideline describes various techniques for gathering requirements



4.3.1.1 Sources of Requirements

Good requirements start with good sources. Finding those quality sources is an important task and, fortunately, one that takes few resources. Examples of sources of requirements include:

- Customers
- Users
- Administrators and maintenance staff
- Partners
- Domain Experts
- Industry Analysts
- Information about competitors

4.3.1.2 Requirements Gathering Techniques

After having identified these sources, there are a number of techniques that may be used to gather requirements. The following will describe the various techniques, followed by a brief discussion of when to use each technique.

To get the requirements down on paper, do one or more of the following:

- Conduct a brainstorming session
- Interview users
- Send questionnaires
- Work in the target environment
- Study analogous systems
- Examine suggestions and problem reports
- Talk to support teams
- Study improvements made by users
- Look at unintended uses
- Conduct workshops
- Demonstrate prototypes to stakeholders

The best idea is to get the requirements down quickly and then to encourage the users to correct and improve them. Put in those corrections, and repeat the cycle. Do it now, keep it small, and correct it at once. Start off with the best structure that can be devised quickly, and expect to keep on correcting it throughout the process. Success tips: Do it now, keep it small, and correct it immediately.

4.3.1.3 Conduct a brainstorming session

Brainstorming is a short group session where everyone is allowed to say whatever they feel is important to the topic of discussion. After that, a facilitator leads the group in organizing and prioritizing the results. The following basic rules for brainstorming ensure better results:

1. Start out by clearly stating the objective of the brainstorming session.
2. Generate as many ideas as possible.
3. Let your imagination soar.
4. Do not allow criticism or debate while you are gathering information.
5. Once information is gathered, reshape and combine ideas.

4.3.1.4 Interview users

Face-to-face contact with users through individual interviewing is the primary source of requirements and an important way to gather and validate their requirements. Remember that it is not the only possible technique, and that interviews can be conducted many different ways. Develop a repertoire of styles to fit different situations. Unless you use the system yourself, you will need to make an effort to understand and experience the user's problem to describe it clearly and correctly.



4.3.1.5 Send Questionnaires

If face-to-face meetings are possible, they are always preferable, because they provide a better means of uncovering the problem behind the problem. Sometimes, though, face-to-face meetings with stakeholders are not feasible (when developing products for the consumer market, for example). In those situations, consider using questionnaires. Send a set of questions, possibly with multiple choice responses, to the relevant stakeholders, and ask them to complete it and return it to you. Success tips: Keep it short and given them a deadline.

This technique has the advantage of providing a lot of information for statistical analysis. However, the questions must be well designed to be clear and to avoid so-called "leading questions", which bias the responses.

4.3.1.6 Work in the target environment

Experience the work of the users for yourself. Working with users helps you understand problems that have resisted previous solutions. Familiar systems developed in this way inevitably include tools for programmers, such as interactive editors and compilers, as the developers naturally have both the expertise in the subject area, and the desire to solve their own problems. It would be good to see the same dedication devoted to solving problems in other areas too. Where the work cannot easily be experienced in this way, it may still be possible to do a bit more than just sit quietly and observe. Users can give you a commentary on what they are doing, what the problems are, and what they would like to have to make the work easier.

4.3.1.7 Study analogous systems

The starting point for many projects is often a similar or an existing system. Sometimes, comparable products and systems contain working versions of good ideas for solving user problems. You can save the time lost in reinventing the wheel by looking at systems already on the market, whether they are systems installed at the user's site or products made by rival organizations. Even if they are trying to solve slightly different problems, they often provide valuable clues as to what you need to do.

Listen when the business asks why a product couldn't do something that the business wants, and keep a list of these suggestions. Later, use it to start discussions with other users. You should be able to obtain some requirements directly this way. If not, capture and store suggestions for future use.

You can describe to users selected features of other products. Explain that the system is designed for another purpose but contains an interesting feature, and you wonder if or something similar would help them. Sometimes these systems are described in documents, such as a contract from another organization or a report written for management. Often, these documents were never intended as formal requirements, and were written merely to communicate a stream-of-consciousness idea. Define a process of going from disorganized to organized information.

Such a process might involve the following activities:

1. Read the document from end to end (several times) to comprehend what the business wants and what actually has been written.
2. Classify all of the types of information in the document. (user, system requirements, design elements, plans, background material, irrelevant detail)
3. Mark up the original text to separate out such requirements.
4. Find a good structure for each of the different types of information such as: a scenario for the user requirements, functional breakdown for the system requirements, and architecture for the design.
5. Organize the information to show gaps and overlaps. Feel free to add missing elements, but confirm these decisions with the users.



6. Create traceability links between these information elements to show the designers exactly what the users want.
7. Convince the business to accept the new information as the basis for the contract.

4.3.1.8 Examine suggestions and problem reports

Requirements can come from change suggestions and user problems. A direct road to finding requirements is to look at suggestions and problems as first described. Most organizations have a form for reporting system problems or software defects. You can ask to look through the reports (and there will probably be many). Sort them into groups so you can identify the key areas that are troubling users. Ask users some questions about these areas to clarify the users' actual needs.

4.3.1.9 Talk to support teams

Most large sales organizations have a help desk that keeps a log of problems and fixes, and support engineers who do the fixing. Many organizations have similar facilities to support their own operations. Talking to the help desk staff and the support engineers may give you good leads into the requirements, and save you time. Also talk to the training team and installation teams about what users find to be difficult.

4.3.1.10 Study improvements made by users

This is an excellent source of requirements. Users of a standard company spreadsheet may have added a few fields, related different sheets together, or drawn a graph, that exactly meets their individual needs. You need only ask: Why did you add that? Their answers help you get to the heart of the actual requirement. This applies also to hardware and non-computer devices. For example, a lathe operator may have manufactured a special clamp, or an arm that prevents movement of the tool beyond a certain point. Any such modification points to something being wrong with the existing product and makes it a valid requirement for the new version.

4.3.1.11 Look at unintended uses

People often use things for purposes for which they were not designed. This is a good way to get new ideas and to think of innovations. For example, an observant product manager noticed that an engineer was staying in the office late to use an advanced computer-aided design system to design a new kitchen layout for his home. Inexpensive commercial products are now widely available for home use.

4.3.1.12 Conduct workshops

Workshops can rapidly pull together a good set of requirements. In two to five days, you can create a set of requirements, and then review and improve them. If everyone in a workshop tries to estimate the cost and value of each requirement, the document becomes much more useful and cost-effective.

Workshops are quicker and better at discovering requirements than other techniques, such as sending questionnaires. You are bringing the right collection of people together, and getting them to correct and improve on their requirements document.

A workshop is inherently expensive because of the number of people involved, but it saves a large amount of time. If you can define the product right the first time and cut three months off the requirements gathering, the savings could be enormous. The workshop has to be thoroughly organized to take advantage of people's time.

Choose a quiet location for the workshop so that people are not disturbed by day-to-day business. Mobile phones should be discouraged; arrange to take messages externally. Take advantage of informal interactions by choosing a site so that people don't go home at night or go out separately.

The example in Figure 17 below shows the logic of a requirements workshop. Note that the workshop provides the environment in which to apply other requirements-gathering techniques such as brainstorming.

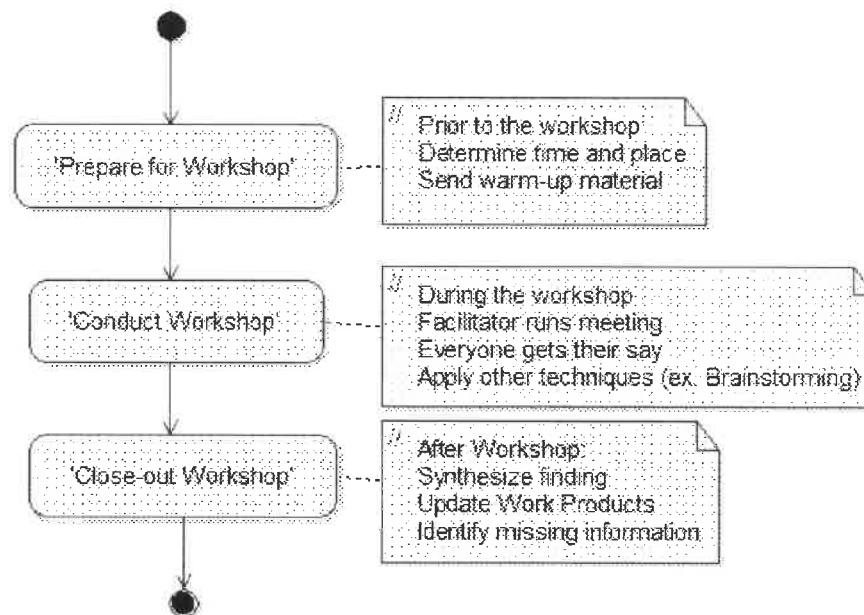


Figure 17—Conducting Workshops

4.3.1.13 Demonstrate prototypes to stakeholders

Prototypes allow us to immediately see some aspects of the system. Showing users a simple prototype can provoke them into giving good requirements information or changing their mind about existing requirements. The techniques described here help you gather ideas for requirements. Prototypes and models are an excellent way of presenting ideas to users. They can illustrate how an approach might work, or give users a glimpse of what they might be able to do. More requirements are likely to emerge when users see what you are suggesting.

A presentation can use a sequence of slides, storyboard, an artist's impression, or even an animation to give users a vision of the possibilities. When prototyping software, make a mock-up of the user interface screens, emphasizing that there is no code and that the system has not been designed or even specified yet (fair warning: there are dangers here for the unwary).

This prototyping aims to get users to express (missing) requirements. You are not trying to sell users an idea or product, you are finding out what they actually want. Seeing a prototype, which invariably is wrong in some ways and right in others, is a powerful stimulus to users to start saying what they want. They may point out plenty of problems with the prototype! This is excellent, because each problem leads to a new requirement.

4.3.1.14 Which Technique to Apply?

Which technique to apply depends on a number of factors, such as:

- Availability and location of stakeholders



- Development team knowledge of the problem domain
- Business's and users' knowledge of the problem domain
- Business's and users' knowledge of the development process and methods

If the stakeholders are not co-located or readily available, for example in the case of a product being developed for mass market, techniques such as brainstorming, interviews and workshops that require face-to-face contact with the stakeholders may be difficult or impossible.

If the stakeholders are available for face-to-face meetings, this is a much better situation and almost all of the techniques described, or combination of them, may be applied. In this case, the domain and development experience of the stakeholders and the development team are critical factors in selecting the appropriate technique.

The figure below provides a framework for determining the appropriate techniques. It defines four main categories of business or user experience and development team experience: "Fuzzy problem", "Selling/Teaching", "Catch up", and "Mature".

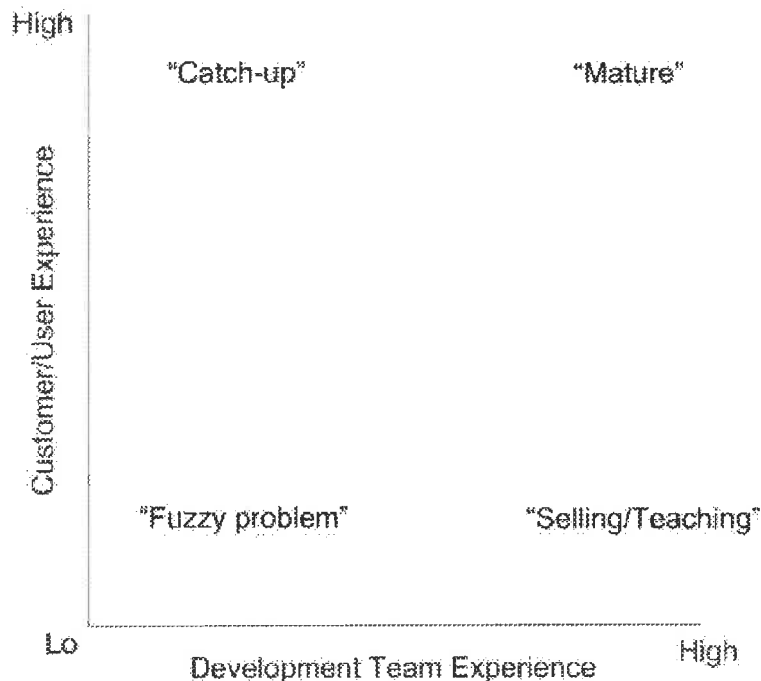


Figure 16----Selection of Workshop Technique

There is no "right answer", but these guidelines may help you decide which method to use:

- *Catch-up*: Interviews, work in target environment
- *Fuzzy*: Brainstorming, workshops
- *Mature*: Questionnaires, workshops, prototypes
- *Selling/Teaching*: Prototypes



4.3.2 Effective Requirement Reviews

This section gives guidelines and discusses how to conduct reviews with relevant stakeholders to ensure agreement, assess quality, and identify changes required.

The cost of correcting errors increases exponentially throughout the project lifecycle. Therefore, it is important to discover problems early enough to solve them quickly and inexpensively.

Requirements reviews are intended to discover problems with the Requirements before you spend significant time and work in implementing the wrong thing. This is not to say that you must have a complete set of requirements before implementation, but be sure to review, internally and with stakeholders, those that are selected for implementation in the early iterations and those that will have a broad impact on the system (often called Architecturally Significant Requirements) to ensure everyone's concurrence before investing significant effort in implementation.

4.3.2.1 Informal reviews

Requirements reviews can be informal, such as simply showing draft requirements to your colleagues or demonstrating a prototype.

These informal reviews are excellent for getting the structure of the requirements correct and removing obvious mistakes. By keeping the review team small, it is easier to make rapid progress. However, informal reviews can miss important perspectives of critical stakeholders.

4.3.2.2 Formal reviews

Requirement reviews can be formal meetings. Start with careful preparation, so that you receive and organize comments before the meeting. The meeting itself should produce decisions on all review items. After the meeting, you must pursue the review actions to completion. If these actions involve a substantial amount of work or require a change to an artifact that is under configuration control, consider submitting Change Requests to prioritize and track the work.

Formal reviews are more wide-ranging and expensive. They provide for more balanced reviews from multiple perspectives. However, formal reviews involve more people, which makes them more difficult to coordinate (thus the need for formality) and expensive in terms of work hours.

4.3.2.3 Two-tier reviews

One technique to get the best of both worlds is to use staged, or "two-tier", reviews. The first tier is informal and performed by a smaller team, possibly many times. The second tier is more formal and performed by the complete group, perhaps only once.

- *First-tier review:* The authors of the requirements and the development team review the requirements during the first-tier reviews to ensure that they are unambiguous, complete and consistent. It is important to include testers and developers to ensure that the requirements are verifiable and feasible. These reviews determine whether the requirements are ready for the larger community to review. First-tier reviews may be informal, formal, or a combination of the two.
- *Second-tier review:* Involve the larger group during the higher-tier review to get more minds working on the problem and to achieve concurrence that the requirements are suitable for implementation and validation. It is best to have one formal requirement review at the Lifecycle Objective (LCO) milestone and, optionally, one at the Lifecycle Architecture (LCA) milestone if significant changes have occurred that introduce unacceptable risk.

Tiered reviews offer several benefits: Eliminate the noise caused by minor edits during the first-tier reviews, allowing subsequent reviews to focus on functionality. Provide a professional look to the requirements, presenting both the requirements and their authors in the best possible light. Safeguard the time of stakeholders who are reviewing the requirements, thus preventing "review burnout", or



diminished effectiveness from overload and stress Provide the best opportunity for full, effective reviews.

4.3.2.4 Golden rules of reviewing

Follow these golden rules of reviewing:

- *Encourage criticism:* Remember that people are improving the requirements, not criticizing you. Even the harshest criticism often contains a grain of truth. Adopt the attitude that every suggestion is a gift.
- *Choose your best reviewers:* A few specific people make the best reviewers, time and again. Cultivate them.
- *Allow adequate time:* It's not over until you have agreed upon and made the corrections.

4.3.3 Qualities of "Good" Requirements

This checklist provides guidance on assessing the quality of requirements

- ☒ *Correct* – Does the requirement correctly specify a true need, desire, or obligation?
- ☒ *Complete* – Is the requirement stated as a complete sentence?
- ☒ *Clear* – Is the requirement unambiguous and not confusing?
- ☒ *Consistent* – Is the requirement in conflict with other requirements?
- ☒ *Verifiable* – Can it be determined whether the system satisfies the requirement?
- ☒ *Traceable* – Is the requirement uniquely identified?
- ☒ *Feasible* – Can the requirement be satisfied within cost and on schedule?
- ☒ *Design independent* – Are all requirements that impose constraints on the design, limiting design options, justified?

4.3.4 Guidelines for Writing Good Requirements

The following is a list of guidelines to follow in specifying decision requirements.

- To write a good requirement, you must write it as a complete sentence, with a subject and a predicate (usually a verb). The subject is an Actor, a stakeholder, the system under development, or a design entity that is related to the requirement. The predicate specifies a condition, action, or intended result that is done for, by, with, or to the subject.
- Consistent use of the verb "to be" solidifies the link between the subject and the predicate. Thus, you can analyze a requirement from a grammatical point of view.
- Beware of lists, bullets, and words such as all, every, and some.

For example:

The order entry clerk must be able to complete 10 customer orders in less than two hours.

This requirement has a subject (the order entry clerk, who is an Actor), a specific and measurable end state (10 customer orders completed), and a performance criterion (in less than two hours).

Follow these simple guidelines in writing any requirement. For consistency, these examples are all in the context of an aircraft.

1. Define one requirement at a time.
 - *The pilot shall be able to control the aircraft's angle of climb with one hand.*



- *The pilot shall be able to feel the angle of climb from the climb control.*
- 2. Avoid conjunctions (and, or) that make multiple requirements.
 - *The navigator shall be able to view the aircraft's position relative to the route's radio beacons.*
 - *The navigator shall be able to view the aircraft's position as estimated by inertial guidance.*
- 3. Avoid let-out clauses or words that imply options or exceptions (unless, except, if necessary, but).
 - *The design shall provide a rear-facing seat for each cabin crew member.*
- 4. Use simple, direct sentences.
 - *The pilot shall be able to see the airspeed indicator.*
- 5. Use a limited (500-word) vocabulary, especially if your audience is international.
 - *The airline shall be able to convert the aircraft from business to holiday charter use in less than 12 hours*

Note: There is no need to use words such as "reconfigured."
- 6. Identify the type of user who needs each requirement.
 - *The navigator shall be able to...*
- 7. Focus on stating what result you will provide for that type of user.
 - *...view storm clouds by radar...*
- 8. Define verifiable criteria
 - *...at least 100 miles ahead.*

4.3.5 Outline Requirements

Early in the project, as part of planning sessions, it may be necessary to define high-level requirements in such a manner as to identify the functional increments and architectural components relevant to the overall project. These are outline requirements.

The purpose of outline requirements is to identify and capture functional and non-functional requirements for the solution. These requirements form the basis of communication and agreement between the stakeholders and the project team on what the solution must do to satisfy project objectives. The goal is to understand the requirements at a high-level so that the initial scope of work can be determined. Further analysis will be performed to detail these requirements prior to implementation

Outline Requirements are typically defined during the Inception Phase, to agree the project scope with the business and stakeholders, and to prepare the ground for Project Planning and Detailing Requirements. The Outline Requirements consist of Decision Requirements and Supporting Requirements.

4.3.5.1 Decision Requirements

The Outline Decision Requirements may address various aspects of the project scope, including:

- the decision points: those points in the business process where decision services are required to make decisions
- the principal decisions to be made at those decision points
- the structure of the decisioning (e.g. dependencies between decisions; re-use of decisions)



- the areas of business knowledge to be harvested, with estimates of their size, complexity and maintenance requirements
- the areas of input data required by the decisioning.

The subsequent tasks – Project Planning and Detailed Requirements Definition – require as a minimum only the decision points and principal decisions; if the other aspects are not addressed at the Outline Requirements stage they can be addressed during Detailed Requirements, using Decision Harvesting. If the business team already has a clear idea of the decision points and decisions, the Outline Decision Requirements task may therefore be limited to documenting these items in an abbreviated version of the Decision Definition Document. Guidance in this approach is provided in Section 4.5.5, Outlining Decision Requirements without DRAW.

Alternatively, if the Decision Requirements Analysis Workshop (DRAW) method is used, all of the above aspects of the scope will be covered in the resulting Decision Definition Document (DDD) with its Decision Requirements Diagram (DRD). This information may be useful for risk mitigation and project planning, but it has a cost: DRAW will typically occupy one or two FICO consultants for a week or two and involve the business SMEs in workshops. Whether DRAW is justified for any particular project is therefore a judgment to be made jointly by the FICO and the project managers, taking into account, for example:

- Do the decision points and decisions remain to be defined clearly?
- Is there much uncertainty around the areas of business knowledge or data required?
- Is the required decisioning substantial and/or complex?
- Are there likely to be multiple interrelated decision services?
- Will this project involve iterative Decision Harvesting and/or development?
- Will development work be divided amongst multiple teams (e.g. for off-shoring)?

(Yes's indicate DRA may be valuable; No's indicate it may be unnecessary.)

4.3.5.2 Supporting Requirements

The remaining Outline Requirements include:

- Functional requirements not directly concerned with decisioning (e.g. the provision of simulation and test environments, bespoke system interfaces, database configuration, reporting)
- Non-functional requirements (e.g. performance, maintenance, reliability, availability and security).
- Other technical requirements

These are documented in the Blaze Advisor Supporting Requirements Document (SRD).

4.3.6 Detail Requirement

The purpose Detail Requirements is to describe system decisioning requirements in sufficient detail to validate understanding of the requirement, to ensure concurrence with stakeholders' expectations, and to permit development to begin.

Having already been outlined in Inception, requirements are typically further detailed in an iterative manner during the Elaboration and Construction phases.

4.3.6.1 Decision Requirements

Decision-based requirements are documented within the Decision Harvesting Workbook.



This topic is extensive. Therefore, it is addressed by itself an entire subsection of this document. See Section 4.9, Decision Harvesting (a.k.a., Rule Harvesting).

4.3.6.2 Supporting Requirements

If there is a need, the remaining requirements are detailed as necessary in the Blaze Advisor Supporting Requirements Document. This includes:

- Functional requirements not directly concerned with decisioning (e.g. the provision of simulation and test environments, bespoke system interfaces, database configuration, reporting)
- Non-functional requirements (e.g. performance, maintenance, reliability, availability and security).
- Other technical requirements

These are documented in the Blaze Advisor Supporting Requirements.

Workshops are an effective way to define both outline and detail requirements or work on a joint application design with the business team.

4.3.7 Identifying Architecturally Significant Requirements

This section gives guideline and explains techniques for identifying and prioritizing architecturally significant requirements. This may be done for both outline and detail requirements and allows managers and architects to track these key requirements.

The identification of requirements that are considered "architecturally significant" is driven by several key driving factors:

- The benefit of the requirement to stakeholders: critical, important, or useful.
- The architectural impact of the requirement: none, extends, or modifies. There may be critical requirements that have little or no impact on the architecture and low-benefit requirements that have a big impact. Low-benefit requirements with big architectural impacts should be reviewed by the project manager for possible removal from the scope of the project.
- The risks to be mitigated: performance, availability of a product, and suitability of a component.
- The completion of the coverage of the architecture.
- Other tactical objectives or constraints, such as demonstration to the user, and so on.

There may be two requirements that hit the same components and address similar risks. If you implement A first, then B is not architecturally significant. If you implement B first, then A is not architecturally significant. Thus these attributes can depend on the iteration order and should be re-evaluated when the order changes, as well as when the requirements themselves change.

Architecturally significant requirements that are poorly understood or likely to change should be prioritized for clarification and stabilization. In some cases, this means further requirements analysis should be done before implementing the requirement. In other cases, some form of prototyping may be best.

4.3.8 FIRUP Guidelines for Conducting a Requirements Workshop

The following guidance is applicable to JAR workshops for both outline and detail requirements. It is also applicable to JAD workshops.

Joint Application Requirements (JAR) or Joint Application Design (JAD) sessions are structured workshops that bring a group of users, analysts and sometimes developers together to jointly define requirements or design a solution. This provides a good alternative to a series of interviews.



JAR or JAD sessions are usually intensive, either full-day or half-day sessions that may go on for a few days or a couple weeks. They need to be managed in a structured manner with a strong facilitator or they will otherwise be frustrating and unproductive experiences for all participants.

An outline of critical aspects for conducting effective JAR/JAD sessions are as follows:

- Planning
- Selecting Participants
- Preparing the Agenda
- Conducting the Session

An effective session will result in:

- Higher quality requirement definition and/or design
- A higher degree of involvement and a stronger sense of ownership
- Shorter requirement and/or design review cycles
- Fewer surprises during User Testing

JAR/JAD sessions help to facilitate an iterative approach towards managing the project and reducing overall risks.

4.3.8.1 Planning a JAR/JAD session

The first thing to understand is the project scope and the scope that is to be discussed for the JAR/JAD session or sessions. It is also important to make sure the Project Sponsor is willing to commit people and perhaps other resources to these sessions. Planning consists of:

- Selecting the location (city, office/hotel, room or rooms): it is usually best to have the session away from distractions, like work.
- Selecting the participants
- Preparing the agenda
- Preparing the participants (provide background on project's current state, business objectives, legacy systems; provide project's objectives, scope, list of deliverables, reference material; conduct preliminary meeting to go over agenda; conduct educational sessions as required; assign pre-work)
- Planning the logistics (projector, computers, internet connections, white board, pens/markers, note pads, name-plates, flip-charts, other session props, snacks/drinks, room layout, computers, printers, rest rooms, travel & accommodations, etc.)

4.3.8.2 Selecting Participants

There are few key roles to have in each session. Here is a short description of the roles:

- Project Sponsor: a representative of management who can keep people focused and motivated towards the end goal. They are also an escalation point for making decisions within the group where consensus is not obtained.
- Facilitator: plays the role of the session leader who does the following:
 - Plans the session
 - Facilitates the session
 - Establishes the "ground rules" for the session



- Makes sure participants adhere to those “ground rules”
- Leads the discussion on issues
- Encourages active participation
- Resolves conflicts that may arise
- Ensures the goals and objectives of the session are met
- Follows up on results/action items

The Facilitator should:

- Have excellent communication and organizational skills
- Possess the ability to negotiate and resolve group conflicts
- Have a firm understanding of the business goals and subject matter
- Be impartial to decisions that are likely to be made
- Not report to any of the participants
- End Users and Managers: these are typically individuals assigned by the Project Sponsor who can represent the business community and effectively communicate business or technical requirements and are given authority to make decisions. There are usually a few or more representatives from each subject area to make sure that multiple perspectives are considered.
- Scribe: keeps records of everything that was discussed during the meeting and quickly publishes them to maintain the momentum.
- Project Leader/Manager: leader of the development team who answers questions about the project's scope, timeline, coordination issues, budget and resources.
- Project Team Members: usually Analysts or Developers from the project team who have expertise in the subject matter or technical solution. These individuals are active to help bring clarity to the requirements and/or design. They may assist the scribe to document requirements/design to maximize accuracy and use outside the meeting.

4.3.8.3 Preparing the Workshop and Agenda

The Facilitator should do the following prior to preparing the agenda:

- Identify the project's objectives, scope, deliverables, constraints, and critical success factors
- Define a sequence and schedule of workshops
- Define expected outcomes and required participant list for each workshop
- Develop the agenda. The agenda identifies the topics to be discussed and the time allocated to each topic. It should contain:
 - The opening: the expectations and desired outcomes and the ground-rules
 - The body: the detailed discussion items, activities and exercises to facilitate participation and keep interest
 - The conclusion: a summary of the session, decisions made, and action items assigned
- Conduct a preliminary meeting to validate agenda and participant list



4.3.8.4 Conducting the JAR/JAD Session

The Facilitator opens the meeting with opening remarks, introductions, and a brief overview of the agenda and objectives for the session. The Project Sponsor should show appreciation for their participation and represent the importance of what is being done.

The Facilitator will direct the session using a prepared agenda. To successfully conduct the session, the Facilitator will:

- Not unreasonably deviate from the agenda
- Stay on schedule
- Ensure that the Scribe is able to keep up with the notes
- Avoid excessive use of technical jargon, so everyone understands the meaning
- Resolve any conflicts
- Provide time for breaks
- Encourage group consensus
- Encourage wide participation so that no one person dominates the discussion
- Ensure that participants abide by the ground-rules

4.3.8.5 Concluding the JAR/JAD Session

At the conclusion of the JAR/JAD session, the Facilitator should confirm that the objectives have been met. In most cases this should mean that the requirements or design documents have been distributed and approved.

4.3.8.6 Keys to Success

Not all JAR/JAD sessions are successful, some are even disastrous. These guidelines are based on research results and personal experiences and are given to avoid failures. They represent the critical success factors and they are to be followed carefully. These guidelines are summarized below:

- Use experienced and skilled facilitators
- Get Executive Sponsor's commitment and support
- Get the right people to participate, predefine their roles and responsibilities
- Set clear defined, well understood and obtainable goals or objectives
- Plan detailed agenda and stick with it
- Define deliverables clearly in advance
- Keep technical jargon to a minimum
- Produce Final Document Quickly

4.3.9 Traceability of Decisioning Requirements

This section provides guidance on how decision-based requirements must be traceable through both the project's life-cycle and through the project's artifacts. The primary goal of traceability is to be able to demonstrate to the business and stakeholders that all the agreed requirements were delivered.

When delivering any sort of functionality to a stakeholder (a rule, custom code, a configured application, a report, a model, a workflow, etc.), the project team must prove that they have satisfied the



requirements, in that they have been designed and tested. The project manager should ensure there is bi-directional referencing between requirements, design and test cases, such that:

- given a requirement, it can be shown where it was designed and tested
- given a test case, it can be shown what requirement(s) or design component(s) it proves

The FIRUP methodology leaves it up to the Project Manager, under consultation with the project team, to determine just how they go about this, but implicit is the need to uniquely identify each testable requirement and/or design component. Some suggestions include:

- a traceability matrix
- making explicit references in the Test Cases as to what requirements and/or design components are referenced
- leveraging an application similar to Rational's RequisitePro, or HP Quality Test Director

4.3.9.1 Overview of Traceability

Traceability is about understanding how high-level requirements (objectives, goals, aims, aspirations, expectations, needs) are transformed into low-level requirements, how they are implemented, and how they are verified.

Using traceability can provide the following benefits:

- Greater confidence in meeting objectives
- Establishing traceability engenders greater reflection on how objectives are satisfied.

Traceability permits coverage analysis to ensure that everything you have done everything that you agreed to do and only what you agreed to do.

- *Ability to assess the impact of change* – Traceability permits various forms of impact analysis that can be used to assess the impact of a proposed change on the cost, schedule, and technical aspects of the project.
- *Improved accountability* – Traceability provides greater clarity about how work contributes to the whole and the ability to track progress
- It is notoriously difficult to measure progress when all that you are doing is creating and revising artifacts. Traceability processes allow precise measures of progress, such as: Is there a design artifact for each requirement? Is there a test case for each requirement?
- *Ability to balance cost against benefit* – Relating product components to the requirements allows you to compare benefits to costs.

Traceability is achieved principally by tracking requirements from definition, through design, development, deployment and functional testing. The principle goals of functional testing are to demonstrate that:

- Each required decision point has been implemented as one or more decision services
- Each required decision has been implemented as defined in the Decision Harvesting Workbook.

Functional testing for a decision service consists of presenting test cases to the service and checking that the returned decision values match those which were expected. A set of test cases therefore has to be defined, with associated expected results. The expected results can be mapped to the requirements through the consistent use of decision names and rule identifiers. The set of test cases should be detailed and comprehensive enough to demonstrate that all agreed upon decision-based requirements as defined by the Decision Harvesting Workbook have been met.

However, traceability extends further than this: successful project delivery and subsequent auditability depends on the ability to trace requirements through all project artifacts and all project activities: outline requirements, detailed requirements, design, and implementation. Some important lines of traceability are shown in the following figure. These are not intended to be complete or authoritative; the approach to traceability in any particular project would be decided by the FICO and the project managers.

The figure below depicts the interrelationship between the various FICO work products and how they work together to ensure the traceability of requirements. (Dashed boxes and lines denote items that are optional under FIRUP.)

For each line of traceability, we wish to show that every requirement at the source of an arrow has been satisfied in the artifact at the target end of the arrow. This is achieved by references back from the target to the source explaining how the requirements have been satisfied. So, for example, there should be a table in the Rule Architecture Design Document (RADD) showing how the decisions defined in the Decision Definition Document (DDD) and in the Decision Harvesting Workbook have been implemented as ruleflow tasks and functions.

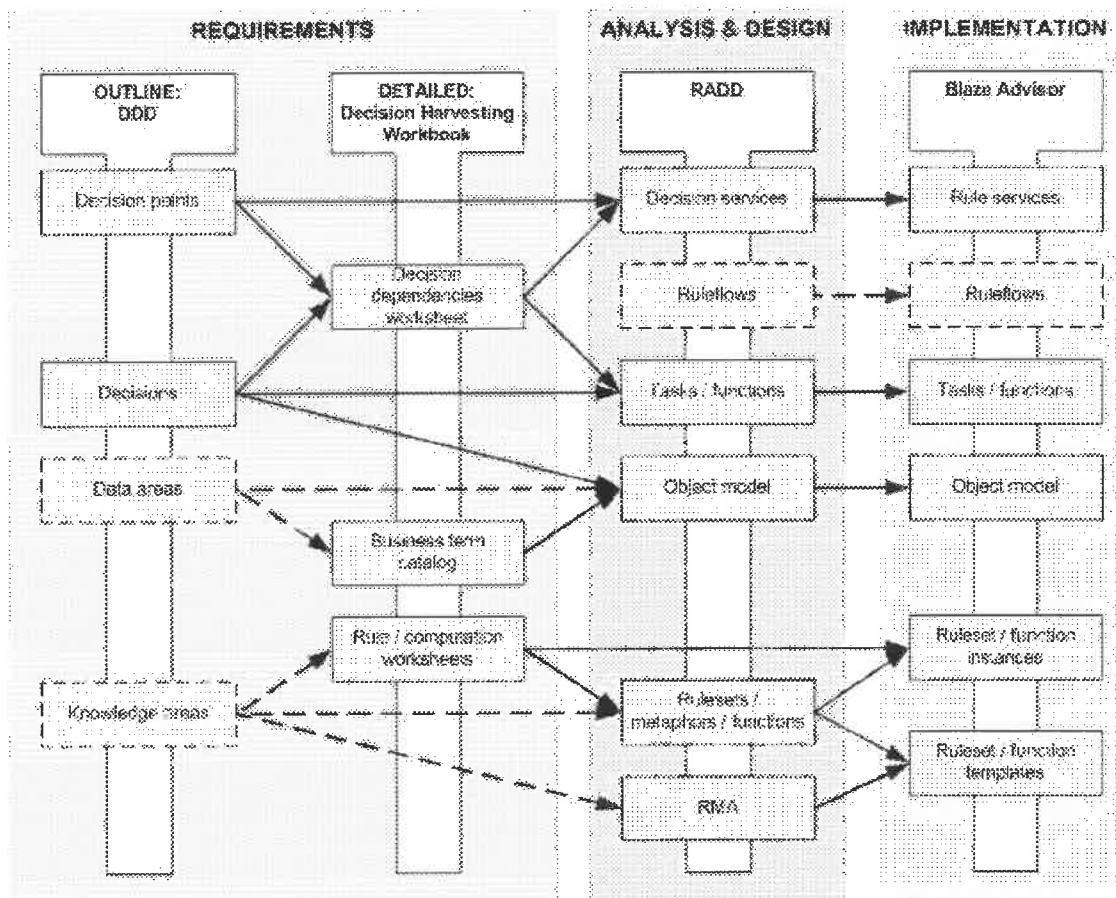


Figure 19—Traceability of Requirements across FIRUP Work Products

Suggestions on how to achieve requirements traceability are provided below.



4.3.9.2 Outline Decision Requirements

The decision points and decisions defined in the DDD should all appear in the Decision Dependencies Worksheet

- The RADD should explain how the decision points defined in the DDD are to be implemented as decision services, and how the decisions defined in the DDD are to be implemented as ruleflow tasks and functions (there may not be a one-to-one mapping in either case)
- Annotations in the object model relate output data items to the decisions defined in the DDD.
- If data areas are defined in the DDD, the Business Term Catalogue should state for each input business term which data area it belongs to, and annotations in the object model should identify any high level structures corresponding with data areas
- If knowledge areas are defined in the DDD, the Harvesting Workbook should state how worksheets relate to them, and the RADD should explain how knowledge areas are implemented as rulesets or functions, and describe their associated RMAs.

4.3.9.3 Detailed Decision Requirements

- The RADD should explain how the decision points defined in the Decision Dependencies Worksheet are to be implemented as decision services, and how the related decisions are to be implemented as ruleflow tasks and functions (there may not be a one-to-one mapping in either case)
- The annotations in the object model should map all input and output data elements to items in the Business Term Catalog
- The RADD should explain how the controlling architecture for the decision logic contained in Rule / Computation Worksheets will be implemented as rulesets, rule metaphors or functions, and describe their associated RMAs.
- Each implemented instance of a rule or function should be traceable to an item of business logic in the Rule / Computation Worksheets. In the case of rules, the simplest way to achieve this is by using the unique identifier specified in the worksheets.

4.3.9.4 Analysis and Design:

There will be a one-to-one correspondence between many items specified in the Rules Architecture Design Document and items of Blaze Advisor code which implement the design.

4.3.10 Review and Approve Requirements

This task describes the process for undertaking a review of the requirements with the various stakeholders and securing formal approval of those requirements. The purpose is to ensure Business and Technical Project Team concurrence to the deliverable requirements.

All requirement work products should be reviewed with the business. The Project Manager will determine the review and approval path for each work product created during the project. This should be documented within the Project Definition Document or Quality Assurance Plan. At a minimum, each work product will be reviewed internally by the Project Manager.

If a work product is to be reviewed externally with the business, it will first go through an internal review. The Project Manager should make a first pass to ensure quality and that the team is prepared with answers to anticipated questions. The Project Manager will schedule and facilitate the review with the appropriate business and technical stakeholders. The Project Manager and author will be present to answer any questions that arise during the review.

A project utilizing Phase and Iteration planning as per FIRUP may have multiple requirement review sessions as additional layers of the requirements are defined



On Blaze Advisor Implementations, The Decision Harvesting Workbook (template) and Blaze Advisor Supporting Requirements (template) should be reviewed and approved

4.3.10.1 Steps

The following are the high-level steps in requirements review and approval process

1. *Schedule and prepare for a requirements review session*

The Project Manager will identify the key stakeholders involved with the review and schedule a meeting. The Project Manager will review the requirement work product and the author will make any refinements as necessary. In advance of the meeting, the author will distribute copies of the requirements to the appropriate stakeholders per the Communications Matrix or as directed by the Project Manager

2. *Conduct Review Meeting*

During the meeting, review each requirement as appropriate, providing particular focus on those items that may have changed from previous sessions

3. *Document review meeting minutes*

Using the Meeting Minutes template, document any agreed upon requirement changes and agreements. Requirements should be agreed to prior to being implemented. Some changes may result in the need to invoke the Change Management process.

4. *Update Traceability Matrix*

Upon completion and approval of the initial requirements documents, transfer the appropriate items to the Requirements Traceability Matrix

4.4 Decision Requirements Analysis

Decision Requirements Analysis (DRA) is a formal method for defining Outline Decision Requirements. First the required business process, decision points and principal decisions are defined then the principal decisions are decomposed into a Decision Requirements Diagram (DRD): a network of sub-decisions, knowledge areas and data areas. This is primarily carried out in Decision Requirements Analysis Workshop (DRAW) sessions, followed by off-site analysis and documentation. It results in the Decision Definition Document (DDD) containing the business process flow, the DRD with supporting definitions, and a statement of project scope

4.4.1 Overview of DRA

Decision Requirements Analysis (DRA) is a method for defining the Outline Decision Requirements for decision services, based on an analysis of the information required for the decisions which they make. The purposes of DRA are:

- to provide a graphical presentation of decisioning requirements to aid discussions with the business and implementation teams
- to minimize risk by providing a precise definition of project scope as early as possible, including decisioning, business knowledge and data requirements
- to support the definition of increments for project planning and the division of work between implementation teams
- to provide analysis of the decisioning structure to aid the design process.

DRA includes the following components:



- The workshop procedure DRAW
- Work Product formats
- Guidance on the use of these outputs in project planning, decision harvesting and design.

DRA can be used in two different contexts:

1. A limited DRAW can be used as an assessment tool to identify the scope for a proposal.
 - If a requirement looks likely to be extensive and complex, DRAW can be carried out as a separate piece of consultancy, to establish the scope of subsequent development projects.
 - Its principal use is for project scoping in the early stages of a Blaze Advisor decision service implementation project.
2. When used under FIRUP for Outlining Requirements, DRA normally takes place during the Inception phase. The results may then be used in:
 - *Project Planning*: to define the scope of functional increments
 - *Detailing Requirements*: to plan Decision Harvesting and set up the Decision Harvesting Workbook
 - *Design*: to guide the design process and provide a set of high-level requirements items for design traceability.

4.4.2 Inputs to DRA

The SOW is a standard input to this process but most of the input is verbal, provided in the workshops. The business SMEs and IT staff may also provide useful documents containing example business rules, data specifications etc.

4.4.3 Outputs Produced from DRA

All the outputs of DRA are recorded in the DDD, described in Decision Definition Document. This document defines the functional requirements for decisioning, provides a graphical representation of decision structure, allows a detailed definition of scope, and identifies all the business knowledge and data requirements. It contains the following:

- A description of the business process, specifying the decision points required and the business decisions to be made at those points
- The Decision Requirements Diagram (DRD): a diagram showing all the information required for the business decisions, broken down into sub-decisions, areas of business knowledge and data
- Detailed functional definitions of all the decision, knowledge and data nodes on DRD
- A statement of scope (defined as sets of nodes on the DRD).

This document (especially the DRD) is used in subsequent tasks:

- Functional increments and/or team work assignments may be drawn on the DRD for Project Planning
- Knowledge areas on the DRD may be used to set up a provisional skeleton Decision Harvesting Workbook for Decision Harvesting

4.4.4 Preconditions and Triggers

DRA can commence immediately after Project kick-off, and is often carried out in same site visit as the kick-off meetings.



4.4.5 Resources and Responsibilities

The roles and staffing for DRAW sessions are described in detail in DRAW. The following staff should be available for the on-site workshops:

- Project Manager (part-time),
- Decision Analyst (full-time),
- Scribe (full-time),
- Technical Architect (part-time)
- Project Sponsors (part-time),
- Operational and Subject-matter Experts (full-time),
- Business Analysts (full-time), IT Analysts (part-time).

The off-site work analysis and documentation following the DRAW sessions will be carried out principally by the Decision Analyst.

4.4.6 Preparation

In advance of the workshops, the Rules Analyst should collect and review:

- the proposal, contract and SOW
- existing requirements documentation
- descriptions of the existing business process and plans for change.

4.4.7 Process Steps

DRA for Outlining Decision Requirements consists of the following steps.

1. Conduct DRAW sessions on site with business. This process fully described in DRAW, and varies project by project, but in summary a typical set of DRAW sessions might include:
 - Workshop 1: Describe the business process flow and define the required decision points and the principal decisions to be made at each decision point
 - Workshop 2: Decompose the principal decisions by defining their information requirements, characterized as sub-decisions, knowledge and data, and create the preliminary DRD
 - Workshop 3: Address each node on the DRD in detail to confirm and elaborate requirements
 - Workshop 4: Presentation of results and preliminary agreement on scope.

Depending on the dynamics it may slow progress to do all the documentation within the workshops. It may be prudent to gain agreement to the framework, take notes on additional information, complete the DRD/DDD outside the workshop and then confirm them in the next available session.

2. Carry out any remaining detailed analysis off site, redraw the DRD, and document all results in the DDD.
3. Deliver, review and sign-off the DDD, signifying agreement on scope



4.5 Decision Requirements Analysis Workshop

Decision Requirements Analysis Workshop (DRAW) is a technique for producing the DRD and its associated documentation in the DDD. DRAW allows the functional requirements for the decision services to be defined in detail, and exposes the knowledge and data sources required. It does not in itself make any design decisions, but does provide the structural information to support those decisions.

4.5.1 Resources

No special facilities are required by way of accommodation or software. The workshops can be held in any quiet meeting-room with a flip-chart or whiteboard. Interaction is improved if the participants sit around a table, rather than all sitting facing the leader as in a presentation.

It would be useful to have a software tool specifically designed for creating DRDs, collecting the associated information, and validating the results, but no such tool exists at present. However, the DRD can easily be sketched on the board during the workshops, and drawn out formally using any commonly available diagram editor (like Microsoft Visio).

The success of DRAW is entirely dependent on the caliber and commitment of the staff involved, so it is important that the right team is assembled for the workshops.

4.5.1.1 FICO Staff

Although DRAW can be carried out by a single consultant, it is better to have a second person as scribe. It is recommended that the following staff be available at least part-time:

- Project Manager: responsible for project planning, management of FICO staff and status reporting
- Rules Analyst: responsible for defining functional requirements
- Technical Architect: responsible for defining technical requirements and technical architecture

The Rules Analyst will lead the workshops, conduct the analysis and produce the outputs. The Technical Architect may be present depending on the needs of the engagement, but must attend any meetings concerning data sources and interfaces. The Project Director and/or Manager should be present for the initial meeting and for any decisions on scope.

4.5.1.2 Client Staff

The client should commit to making the following staff available at least part-time:

- Project Sponsors: ultimately responsible for the business success of the project
- Project Manager: responsible for day to day project management, liaison and reporting
- Operational Experts: contributing subject-matter expertise; instrumental in driving and fostering user acceptance and involvement
- Business Analysts: responsible for business design and definition of functional requirements
- IT Analysts: providing IT support, e.g. interface development, hardware configuration, environment preparation

Most of the input will come from the Project Sponsors, the Operational Experts and the Business Analysts. IT Analysts should be available to support the workshops and analysis as required, e.g. with data mapping and legacy systems expertise.



4.5.2 Scheduling

DRAW works best in small groups or in individual interviews, rather than large workshops. Start with one or more preliminary workshops to draw out the top-level requirements then pursue more detailed analyses with sub-groups or individual specialists as necessary. Depending on the dynamics it may slow progress to do all the documentation within the workshops. It may be prudent to gain agreement to the framework, take notes on additional information, complete the DRD/DDD outside the workshop and then confirm them in the next available session. After the results have been analyzed and documented, the draft DRD/DDD should be presented at a final plenary session for general agreement.

It is not necessary to explore the whole network with each person: individual areas of expertise or responsibility may correspond to different regions of the network. It is however a good idea to cross-check results with multiple staff: individuals will miss items and there may be differences of opinion which reveal important issues which need to be clarified.

Be flexible. It is unlikely you will be able to conduct all the interviews according to a pre-arranged schedule; as the network is revealed you may find it useful to contact and interview additional people.

4.5.3 Technique

In principle, DRAW consists of a standard sequence of stages:

- Identify the decision points within the business process flow
- Decompose the decisions using DRA to produce a DRD
- Define the scope using boundaries on the DRD

These stages are briefly described in following sections. It should be noted that what is described is an ideal approach and that the approach may need to be adjusted to optimally fit the given situation.

Of course, no real workshop will run quite this smoothly: things will emerge in later stages or over multiple workshops, which will require you to backtrack and modify or add detail to the results of previous stages. This is not a problem, so long as the basic principles are followed (for example, any decision, whenever it is added to the DRD, has to be defined and decomposed).

4.5.4 Process

The DRAW process follows the following basic steps:

1. Identify the "Decision Points" within the Business Process
2. Perform Decision Requirements Analysis
3. Define the Scope

4.5.4.1 Identify the "Decision Points" within the Business Process

Start by inviting the business team to describe their business process, and what they hope to achieve from this project. Try to understand the drivers: What transformations does the business team intend to bring about in their business? What benefit do they hope to achieve through Decision Management?

Investigate in detail all the points in the business process where decisions will be required of decision services. Give each decision point a name, and define:

- The stage in the business process where this decision point occurs
- The decision-making carried out at this stage
- The intended role of the rule service in the decision-making
- The role of any users or other system components



For example:

Decision Point: Decide Pre-Bureau Eligibility

Description: Blaze is called after collecting all customer data, to validate the data and filter out easily identifiable ineligible applications before incurring the expense of buying bureau data. If the data are invalid the service will respond with REFER, to allow the data to be manually corrected. If not, it will respond with CONTINUE or DECLINE. No Credit, Fraud or Insurance referrals occur at this point.

Document the results in the DDD, using some sort of business process flow diagram or use case.

For example, see Figure 20 on the following page.

For each decision point, establish the decisions which the business requires to be made by decision services, expressed in the most general possible terms. Wherever similar decisions are made in different business contexts, try to generalize the decisions across contexts. Even if there are expected to be multiple decision services do not at this stage attempt to allocate decisions to particular services, just identify which decisions have to be made.

Give each decision a name, and then establish the following:

- A question which characterizes the decision, expressed clearly in natural language, with a defined set of answers.
- Any other results which are required to be returned with the answer.
- The decision points at which the decision has to be made and any variation in the required decision across the contexts.

For example:

Decision: Eligibility in Principle

Question: Is the applicant eligible in principle for a loan?

Answer: CONTINUE (the requested application may proceed), REFER (the requested application should be referred for manual assessment), or DECLINE (the requested application cannot be approved)

Other results: A list of explanations for the decision

Decision points: Decide Pre-Bureau Eligibility and Decide Post-Bureau Eligibility. At Decide Pre-Bureau Eligibility only CONTINUE and DECLINE are used; thereafter any of the three values may be returned.

Record the business process flow and the definitions of the top-level business decisions in the DDD. All business terms used in defining the decisions should also be recorded and defined in the Business Terms Catalog.



FICO Recommendations Whitepaper for Chubb Business Rule COE

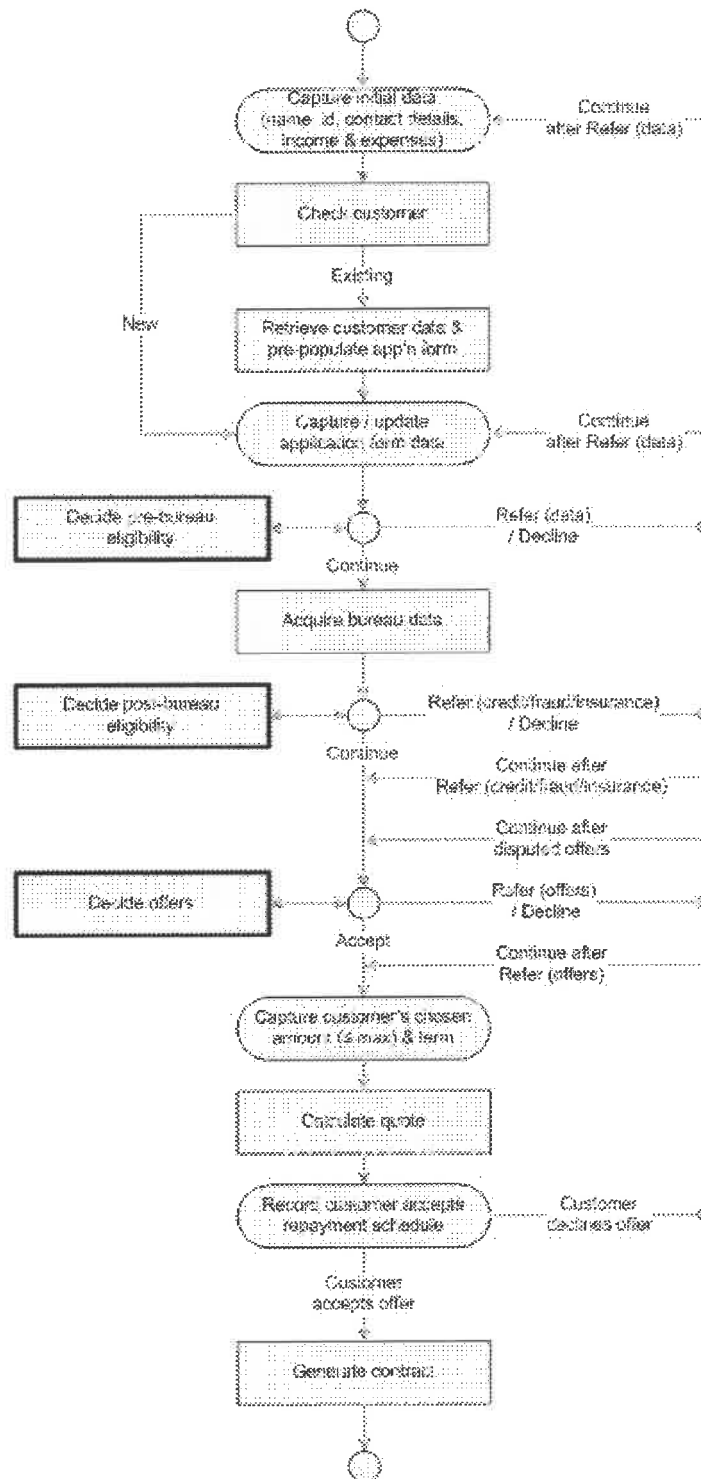


Figure 20—Sample Business Process Flow Diagram

March 31, 2011

FICO Confidential Page 93 of 259

Confidential

FED007129_0093

4.5.4.2 Perform Decision Requirements Analysis

This is an iterative process. Choose any decision which has not yet been decomposed, and ask the business: “What information is required to take this decision?” Fully explore the following possibilities:

- The results of other decisions (sub-decisions)
- Supporting information:
 - Knowledge currently existing as human expertise
 - Guidelines, policies, etc currently existing in documents
 - Specific data describing the case
 - Reference data from databases or other systems

For each required sub-decision, add a decision node to the DRD (labeled with its question) and define it as above. It can then be decomposed in its turn. For each required area of supporting information, decide whether it comprises knowledge or data, and accordingly add a knowledge or data node to the DRD.

For example:

When you ask the business the question “What information do you need to decide whether the applicant is eligible for any offers?” – they might reply “Well, you need to know whether the applicant is eligible in principle, and what offers are available, then apply some rules to decide whether those offers should be made to the applicant” This tells you to add two sub-decision nodes and a knowledge node to the initial starting node. The following example shows how continuing this process can start to generate the DRD in Figure 21:

Decision: Eligibility for Offers

Information Required: Results of two sub-decisions (Eligibility in Principle and Offers), eligibility rules.

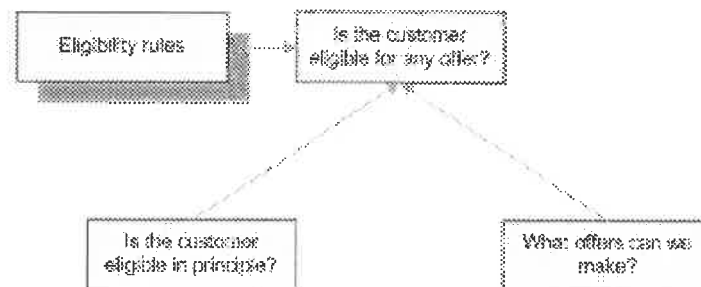


Figure 21—Sample DRD Decision Node



Decision: Eligibility in Principle

Information Required: Results of two sub-decisions (Data Validity and Risk Grade), policy rules, personal details, application form data, income and expenses data, bureau data

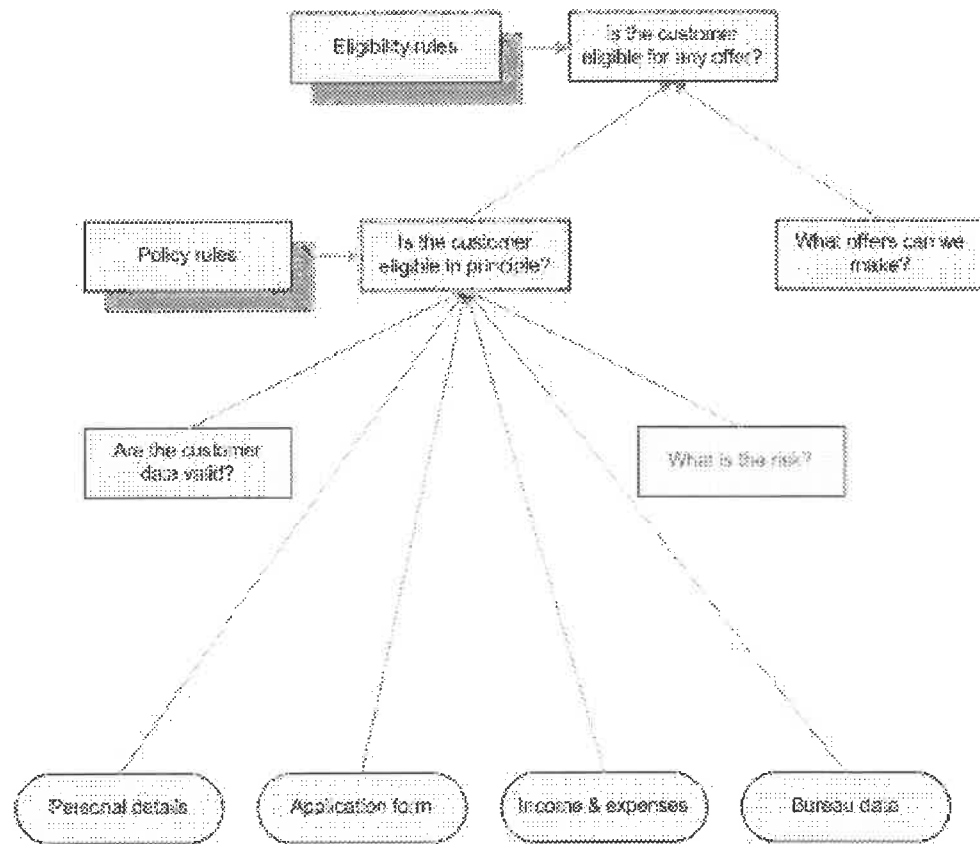


Figure 22—Example Showing Supporting Rules and Data for a DRD Decision Node



FICO Recommendations Whitepaper for Chubb Business Rule COE

Decision: Data Validity

Information Required: Data validity rules, personal details, application form data, income and expenses data.

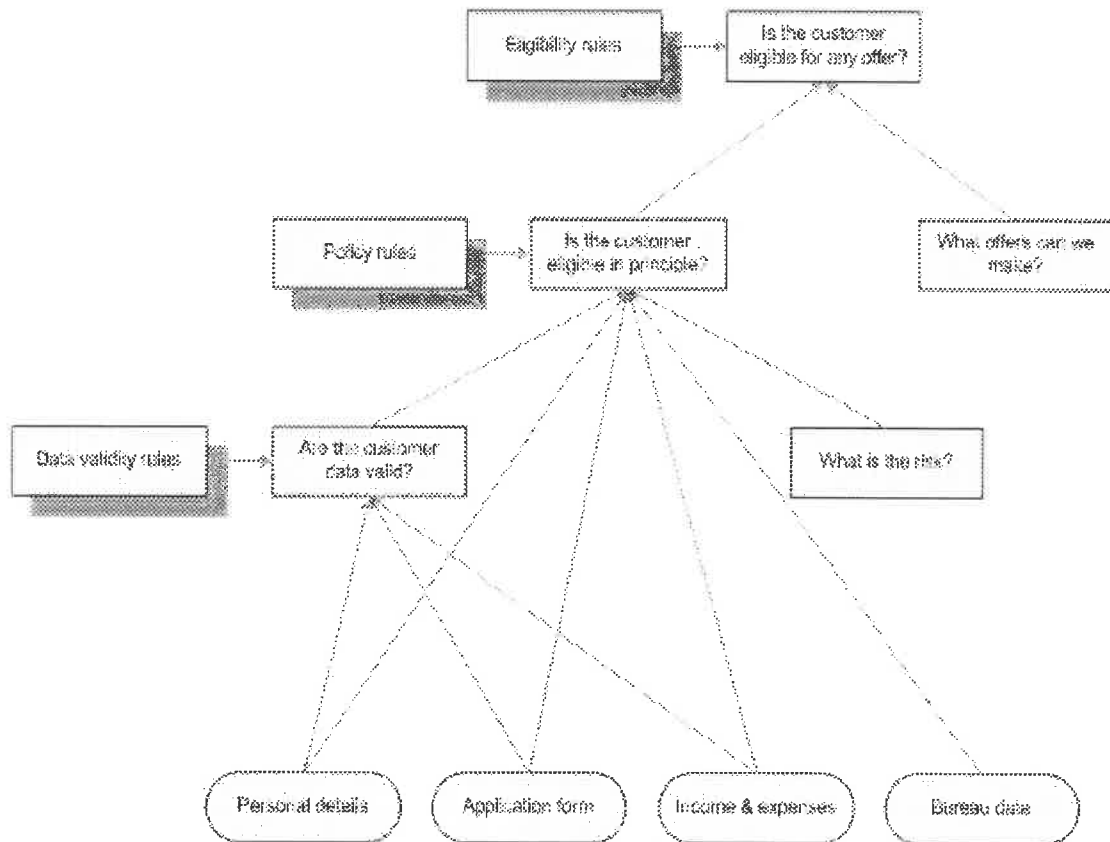


Figure 23—Example Data Shared by DRD Decision Nodes



FICO Recommendations Whitepaper for Chubb Business Rule COE

Decision: Risk Grade

Information Required: Joint Odds Matrices, application risk score (for new customers), behavior score (for existing customers), and bureau data.

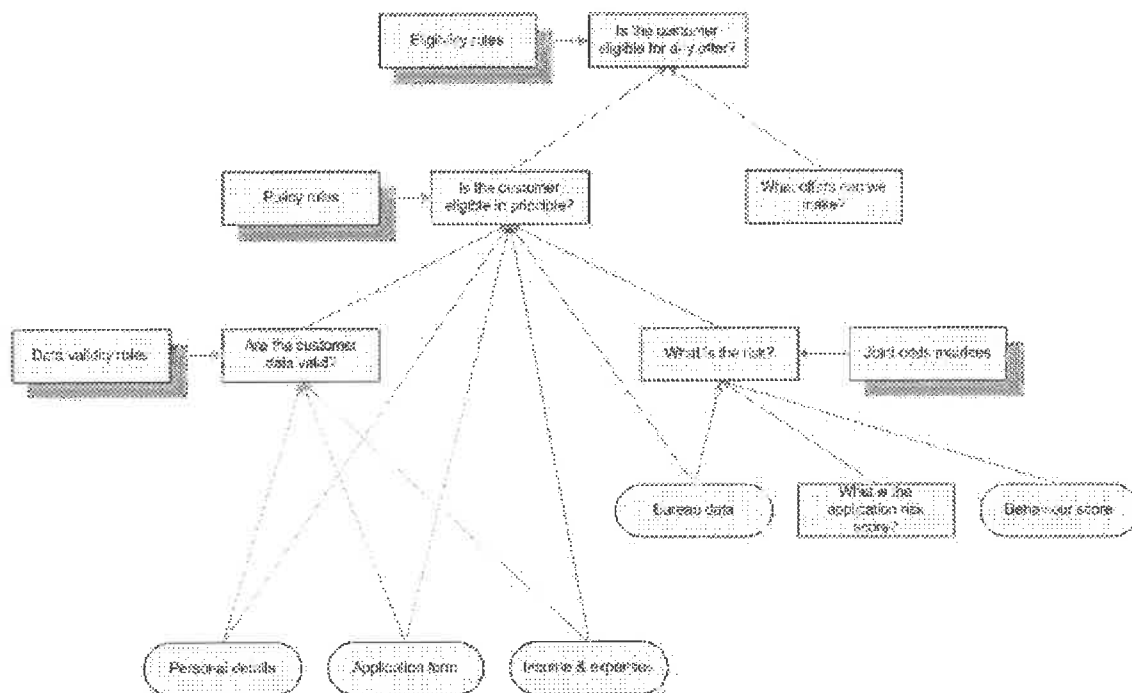


Figure 24—Example Showing Continued Elaboration in DRD

**Decision: Application Risk Score**

Information Required: Application risk scorecards, personal details, application form data, income and expenses data, requested loan.

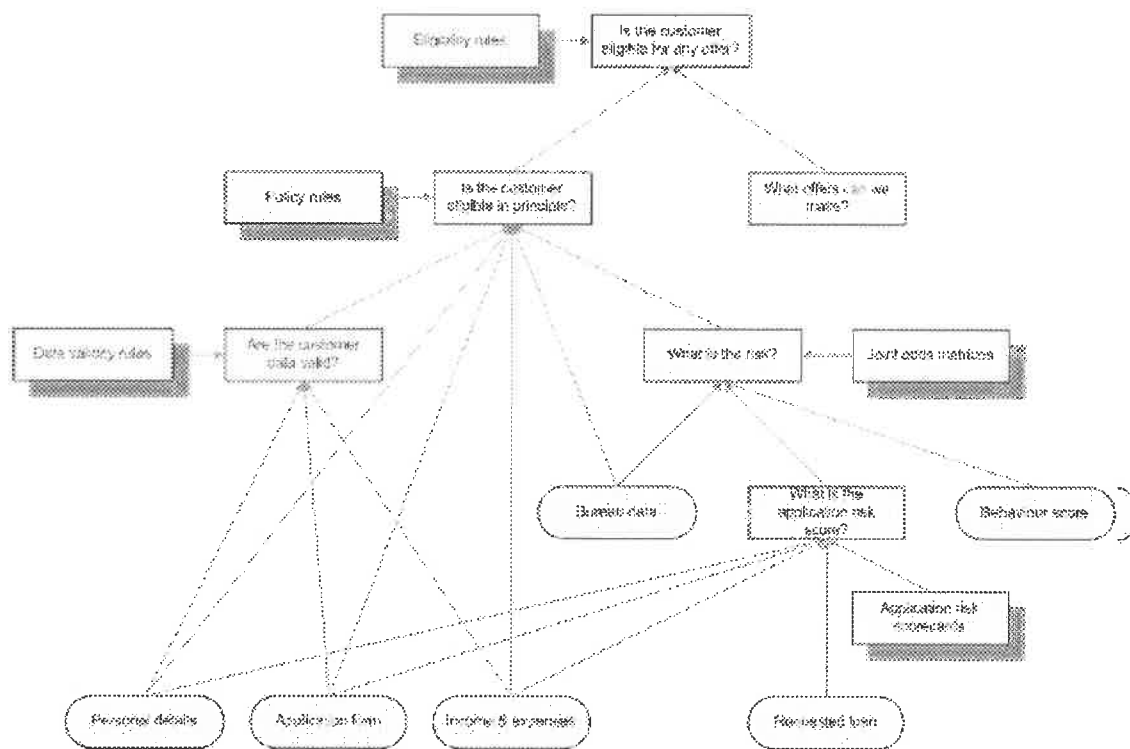


Figure 26—Sample Completed DRD

Note that because this process is recursive, and only analyses one decision at a time, it is not more difficult to decompose a large decisioning requirement: it just takes longer.

When the process is complete you should have a complete DRD, identifying the decisions required, the areas of knowledge involved in the decisions, and the areas of data required by the decisions.

You should then revisit all the nodes in the DRD and discuss them in detail with the business, one by one.

For each decision node, agree on a detailed description which explains how its required knowledge and data are used in reaching decision.

For example:

Decision: Eligibility in Principle

Description: If the data provided by the customer are valid, policy rules decide whether the applicant is eligible in principle for a loan, taking into account the branch data, customer data, bureau data, and the Risk Grade. Each rule will return a REFER or DECLINE decision; DECLINE will override REFER. REFER rules will not apply at the Decide Pre-Bureau Eligibility decision point.



For each knowledge and data node, investigate the information required, and record:

- The source of the information (data systems, documents, or personnel)
- The maintenance requirements (who will be maintaining it, and how often)
- Any other important background, for example the size, complexity, accuracy and completeness of the information source, any possible modifications within the timescale of the project

For example:

Knowledge: Policy rules

Description: About 30 rules to decide on Eligibility in Principle, for example:

- If the applicant's income is less than R1000, the application should be declined
- If the applicant has indicated that they have their own insurance, the application should be referred to Insurance
- If the bureau data indicate that the applicant's ID does not match their name, the application should be referred to Fraud
- If the applicant's risk grade is "Decline", the application should be declined.

Source: These rules will be based on the existing Policy Rules documented for Frontier Credit Application Processing, updated and extended as required by AR Credit.

Maintenance: Maintained infrequently (a few times per year) by business users through an RMA.

Comments: The current authority is Xavian Pert (Product Director) who will be unavailable after 30 September.

Note that at this stage the objective is not to collect actual knowledge (e.g. particular business rules or pages of equations), but just to describe the required areas of knowledge and identify the sources for subsequent discovery. However, it is likely that the business will offer example rules, and if so these should be recorded.

4.5.4.3 Define the Scope

Having defined the structure of the required decisions and the supporting information, decide which areas are to be implemented as part of a decision service and which are to remain as external systems. Do not at this stage try to decide how many decision services there should be and how decisions should be allocated between them, simply identify which nodes are to be implemented and which are not. This will provide you with the "project scope boundary".

For example:

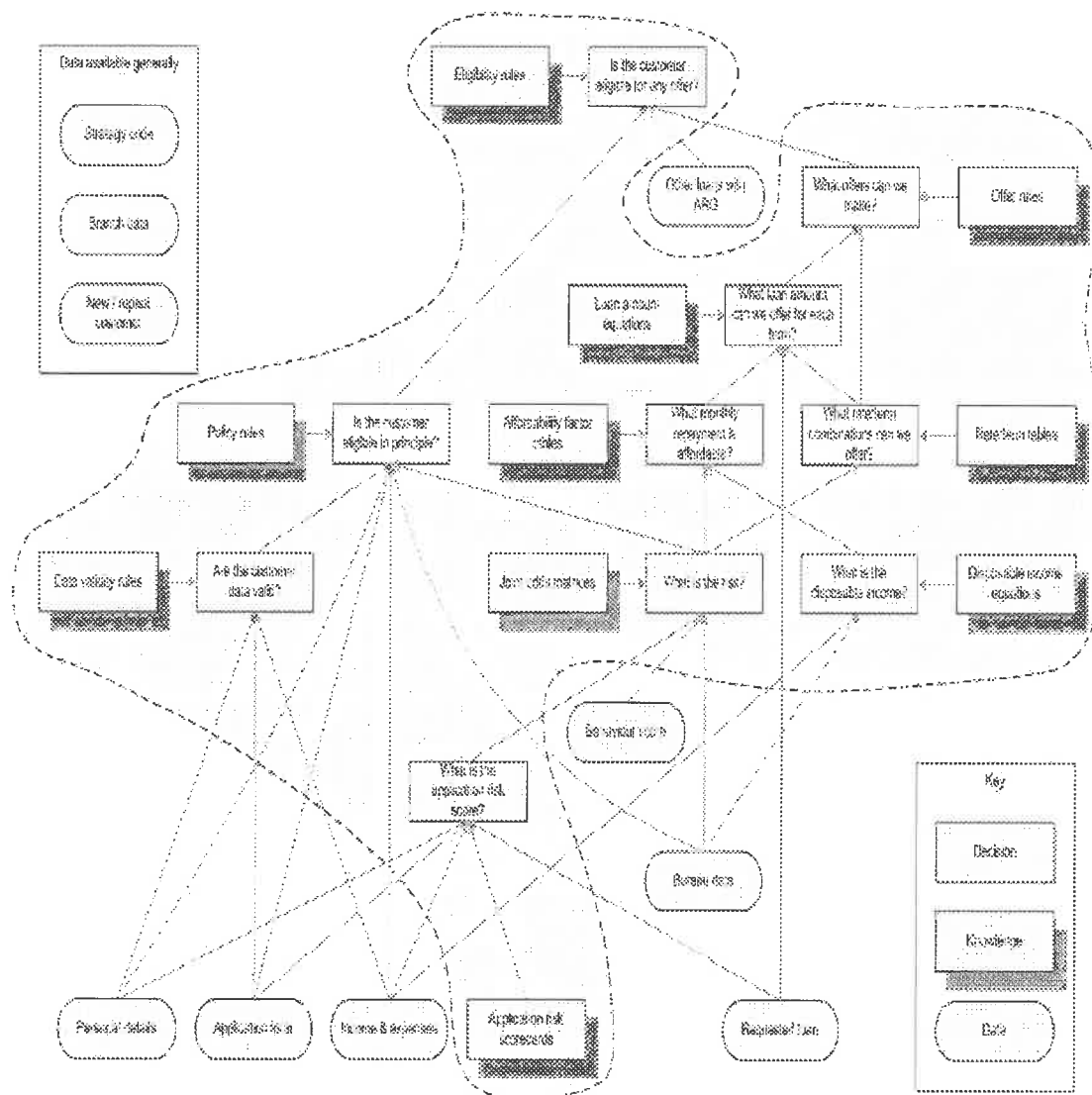


Figure 26—Sample DRD Showing Project Scope Boundary

Some useful rules of thumb are:

- The top level decision nodes will be implemented in decision services (this was the starting assumption).
- Most of the sub-decisions will be implemented in decision services, but decision services may call out to external decision-making processes e.g. web services or human authorities.



- Most knowledge nodes will be implemented in the decision service (as rulesets or functions), but some very simple rules can be implemented as queries on a database.
- Most data nodes will be information sources outside the decision service, but some may be implemented for convenience within the decision service if no other system requires access to them.
- The scope should be defined in the DDD using a boundary drawn on the DRD and a verbal definition.

Boundary analysis involves provisional architectural design decisions, so the boundaries may be subject to change later in the design process. By recording the results of this analysis clearly in the project scope, any such requirements can be made the subject of a change request and properly costed.

4.5.5 Outlining Decision Requirements without DRAW

Outlining Decision Requirements normally takes place during the Inception phase, at or immediately after the project kick-off meeting, with the purpose of defining the scope of decisioning: the decision points and the principal decisions to be made. The results are documented in an abridged Decision Definition Document

4.5.5.1 Scenario Setup

Outlining Decision Requirements normally takes place during the Inception phase. The use of DRA for this is described in Decision Requirements Analysis; this subsection describes the process when DRA is not used.

The purpose of Outlining Decision Requirements is to define the scope of decisioning, including (as a minimum):

- the decision points: those points in the business processes where decision services are required to make decisions
- the principal decisions to be made at those decision points.

If the business already has a clear idea of the decision points and decisions this task may be limited to clarifying and documenting these items. If not, they can be agreed in business process workshops. These results can be documented in an abridged Decision Definition Document., and may then be used in:

- Project Planning: to define the scope of functional increments
- Detailing Requirements: to plan Decision Harvesting.

4.5.5.2 Inputs

The SOW is a standard input that identifies the project scope at a very high level but most input to this process is verbal, provided in the workshops. The business SMEs and IT staff may also provide useful documents containing example business rules, data specifications etc.

4.5.5.3 Outputs

The results are documented in an abbreviated version of the Decision Definition Document, leaving out the DRD and detailed definitions which would result from DRAW but providing the business process flow (optional), decision points, principal decisions and a statement of scope. If any agreements have been reached with the business on the scope of business knowledge to be harvested or input data to be used, these should also be recorded in the DDD



4.5.5.4 Preconditions and Triggers

Outlining Decision Requirements can commence immediately after Project kick-off, and is often carried out in same site visit as the kick-off meetings.

4.5.5.5 Resources and Responsibilities

The following staff should be available on-site for the kick-off meetings and/or business process workshops:

- FICO Project Manager (part-time), Rules Analyst (full-time)
- Client Project Sponsors (part-time), Project Manager (part-time), Operational Experts: (full-time), Business Analysts (full-time), IT Analysts (part-time).

The documentation of the results will be carried out off-site, principally by FICO Rules Analyst.

4.5.5.6 Preparation

In advance of the meetings/workshops, the Rules Analyst should collect and review:

- the proposal, contract and SOW
- any existing requirements documentation
- descriptions of existing business process and plans for change.

4.5.5.7 Process Steps

Outlining Decision Requirements consists of the following steps.

1. Conduct on-site meetings and/or workshops with the business to agree (as a minimum) the business process, the decision points and the principal decisions
2. Document the results
3. Deliver, review and sign-off the document, signifying agreement on scope

4.6 Decision Requirements Diagram (DRD)

The Decision Requirements Diagram (DRD) is a network diagram (technically speaking, a directed acyclic graph) which shows the information required by all the business decisions identified for the Decision Points. It can be thought of as a decomposition of those business decisions into sub-decisions and areas of supporting information: business knowledge and data. The DRD is intended to give an “at a glance” view of the scope and structure of the required decisioning. Detailed definitions of all the nodes on the diagram are provided in the DDD

As discussed in the previous section, the heart of DRA is the Decision Requirements Diagram (DRD). When DRA is used, this diagram is central to the process of Outlining Decision Requirements. It is both the principal output of DRAW and a major input to Project Planning and Detailing Requirements.

The figure below is the simple example of a DRD for a secured loan originations decision taken from the Decision Requirements Example in the previous section. This example is rather simple, but shows all the principal features of a DRD.

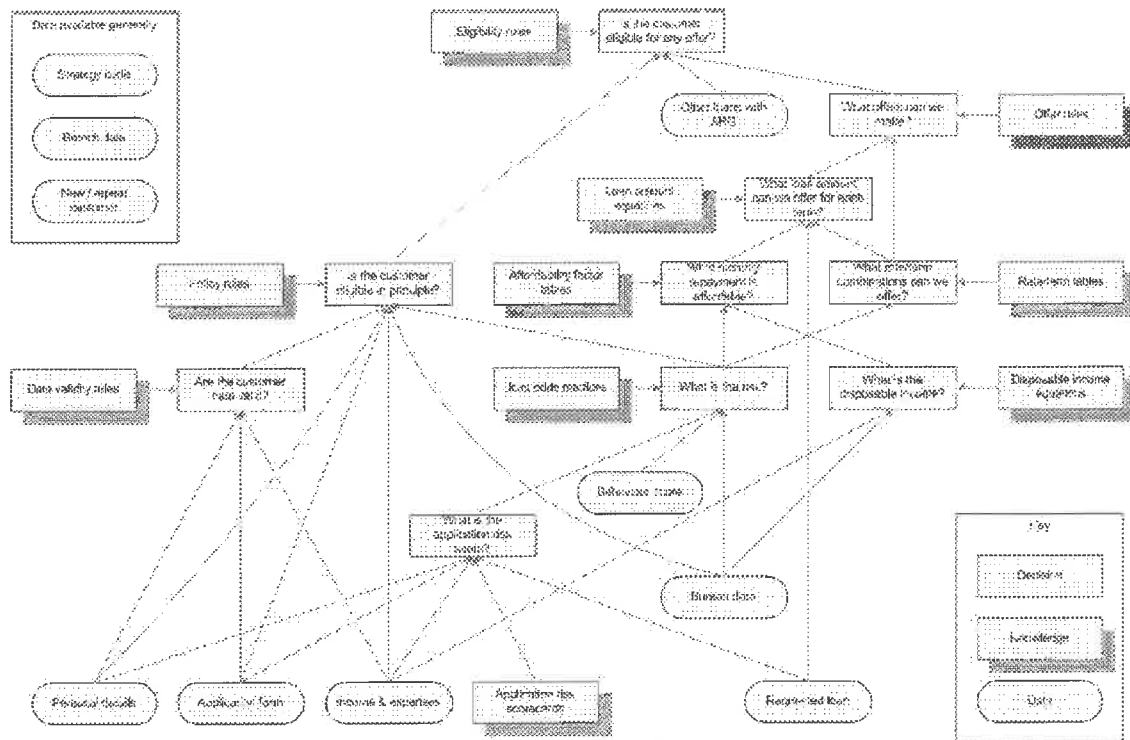


Figure 27—Example Decision Requirements Diagram (DRD)

The DRD contains three types of nodes: decisions, knowledge and data. These are linked by arrows which indicate requirements: an arrow from A to B indicates that A is required for B. For example, in the figure above, the decision “What is the risk?” requires four pieces of information: the result of the decision “What is the application risk score?” business knowledge expressed as joint odds matrices, the behavior score, and bureau data.

By convention decisions are decomposed downwards, so the arrows normally point upwards. Note that the diagram is not necessarily a tree: a single piece of information (e.g. the Income & Expenses data in Figure 27) may be required by multiple decisions.

4.6.1 Components of a DRD

A DRD contains only decision nodes, knowledge nodes, data nodes and arrows.

4.6.1.1 Decision Nodes

Each decision node in the DRD is a decision which is required to be taken, but not necessarily by a rule service: it might alternatively be taken by a human decision-maker or by another system component (e.g. a bureau service or an analytical model). Each decision is characterized as a means of providing an answer to a question, using.

- a name for the decision
- a natural-language question characterizing the decision
- a simple definition of the decision, explaining how areas of knowledge and data are used to determine the result.



4.6.1.2 Knowledge Nodes

Each knowledge node is an area of business knowledge required by one or more of the decisions. This knowledge may currently exist as human expertise, printed documents etc. In a decision-making system such knowledge may be represented formally as (for example):

- Rules: business policies which must be “discovered” and coded in Blaze
- Scorecards: predictive models which must be generated using analytics or defined manually
- Equations or algorithms: precise definitions of numerical calculations.

4.6.1.3 Data Nodes

Each data node represents a set of data required by one or more decisions, which might be:

- Specific data relating to the case being processed
- Reference data held externally (e.g. on databases) or internally (e.g. in decision tables).

Note that for the sake of clarity areas of data which are widely required across many decisions (for example in Figure 27 the applicant details and application details) need not necessarily be linked by arrows; this would make the diagram unreadable. However, these should be shown separately on the DRD, and must be listed in the decision and data node definitions.

4.6.1.4 Arrows

Normally, only decision nodes are decomposed, so each arrow represents the fact that a piece of information is used by a decision. Requirements are not conditional; the arrow must be present if the piece of information is ever required to make the decision.

Arrows are not separately defined in the supporting documentation; the requirements they represent should be covered in the descriptions of the decisions.

4.6.2 Uses of the DRD

The definitions of the individual nodes provide a basis for defining scope:

- Decision nodes provide a clear definition of the functional requirements for decisioning
- Knowledge nodes identify all the areas of knowledge discovery required
- Data nodes identify all the data to be made available and modeled.

Depending on the context, the arrows between the nodes have implications which are important for the design, including:

- Data flow: if the source of some information is external to the system component in which the decision is to be implemented, a data interface is required
- Ordering of tasks: if decision A requires the result of decision B, B must be evaluated before A.

Drawing a boundary around subsets of nodes allows a very clear definition of what is inside and what is outside the boundary, and the lines crossing the boundary indicate the interfaces necessary. This allows:

- Project scoping: by drawing a boundary around all the nodes to be implemented in the project
- Iteration planning: by partitioning the nodes between a number of increments
- Division of functionality: by drawing boundaries between a number of rule services and/or other system components
- Resource allocation: by dividing nodes between teams (e.g. for off-site development).

The graphical nature of the DRD makes it a very useful tool for supporting discussions with the project management and business personnel.

Two examples of the use of boundaries on the DRD are shown below.

The figure below shows the DRD used to define project scope. In this case, all the decisions and knowledge areas are in scope, all the data areas are out of scope (i.e. decision services must implement all the defined decisioning, and all the defined data must be provided by the infrastructure to the decision services).

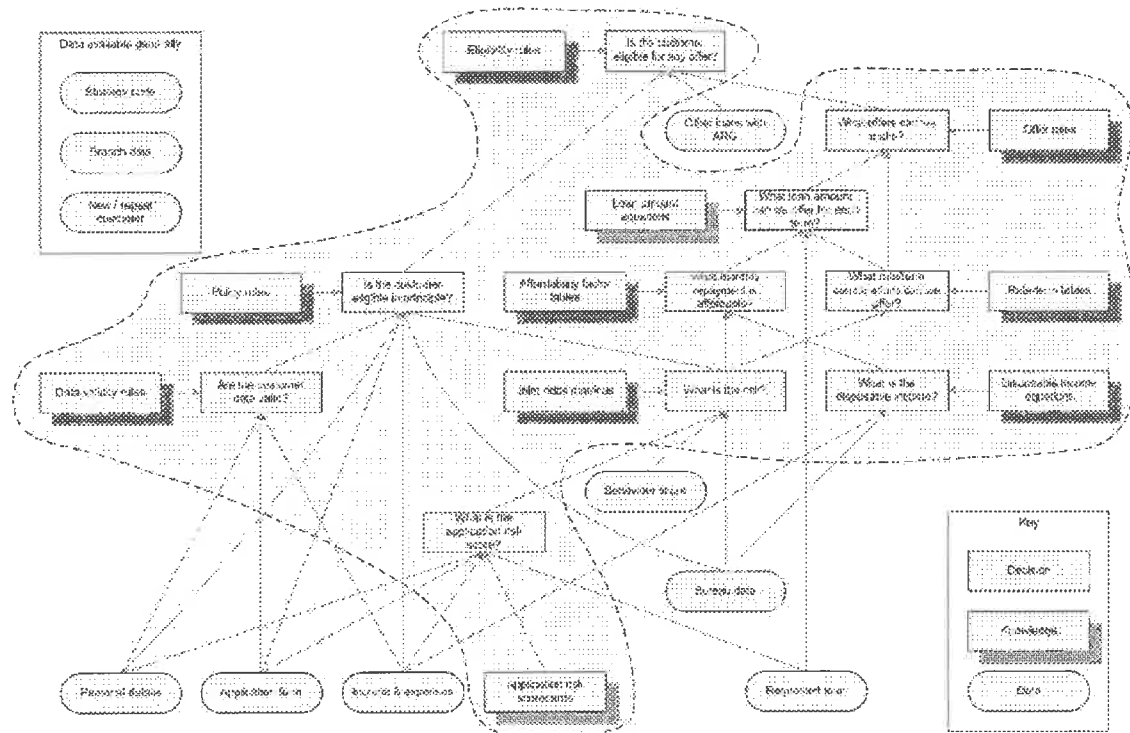


Figure 22—Example Decision Requirements Diagram Showing Project Scope

The figure below shows the DRD used for iteration planning. Here the decisions and knowledge nodes are divided into three increments, each providing a distinct functional benefit. Decision Requirements Analysis has allowed this division to be planned so that the three increments are “loosely coupled”: i.e. connected by few arrows.

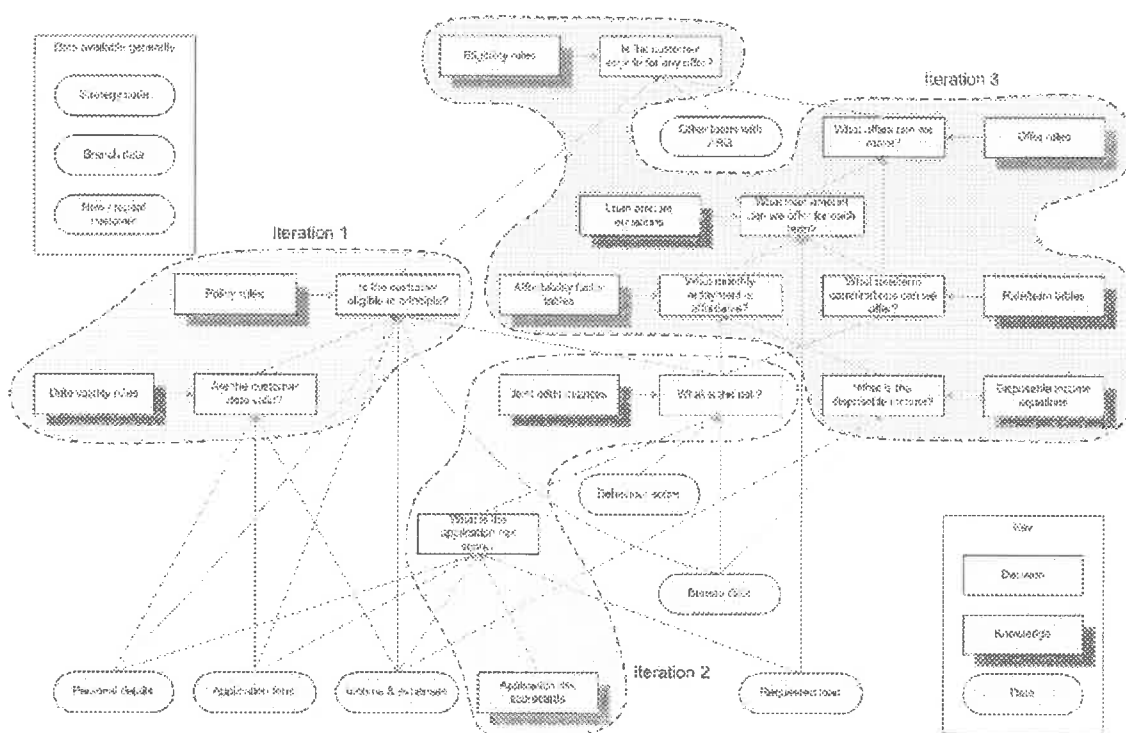


Figure 23—Example Decision Requirements Diagram Showing Increments

4.7 Decision Definition Document (DDD)

4.7.1 Purpose and Scope

The Decision Definition Document (DDD) defines the functional requirements for decisioning, provides a graphical representation of decision structure, allows a detailed definition of scope, and identifies all the business knowledge and data requirements.

It contains the following:

- A description of the business process, specifying the decision points required and the business decisions to be made at those points
- The Decision Requirements Diagram (DRD): a diagram showing all the information required for the business decisions, broken down into sub-decisions, areas of business knowledge and data
- Detailed functional definitions of all the decision, knowledge and data nodes on DRD
- ⊗ A statement of scope (defined as sets of nodes on the DRD).

4.7.2 Description of Major Sections

The DDD is four main sections:



- The Business Context section (see 4.7.2.1) describes the background to the project including the business goals and the intended role of the Blaze Advisor decision services.
- The Decision Points section (see 4.7.2.2) describes the decision points in the business process from which Blaze Advisor decision services will be called, and describes at a high level the decisions required at each decision point.
- The Decision Requirements section (see 4.7.2.3) provides a detailed definition of all the decisions to be taken by Blaze Advisor decision services, noting the areas of business knowledge involved and the data required.
- The Scope section (see 4.7.2.4) defines the scope of the implementation project by referring to the definitions in the Decision Points and Decision Requirements sections.

4.7.2.1 Business Context

4.7.2.1.1 Background

The Blaze implementation project will normally be part of a larger program designed to achieve certain business objectives. This section should describe this larger context, explaining the business objectives and the intended role of the decision services.

4.7.2.1.2 Objectives

This section should list explicit objectives for the Blaze implementation.

4.7.2.1.3 Assumptions

This section should list any assumptions underlying the DDD.

4.7.2.2 Decision Points

This section should describe the business process flow, using both a verbal description and a diagram. It should identify within the business process flow all the *decision points*: points where business decisions need to be made by Blaze decision services.

Each identified decision point should be defined in its own sub-section, covering:

- *Name*: a verb phrase beginning with "Decide..."
- *Description*: a high-level description of the decision point and an explanation of the context of the business decisioning
- *Decisions*: a list of the business decisions which need to be taken at this point
- *Responses*: a high-level description of the results provided by the decisions

All the business decisions identified in this section must be defined in more detail in the Decision Requirements section and must appear on the DRD.

4.7.2.3 Decision Requirements

This section provides the results of analyzing the information requirements for the business decisions identified for the decision points using Decision Requirements Analysis (DRA). It contains:

- The Decision Requirements Diagram (DRD)
- Definitions of all the decision, knowledge and data nodes on the DRD.

4.7.2.3.1 Decision Requirements Diagram

The Decision Requirements Diagram (DRD) is a network diagram (technically speaking, a directed acyclic graph) which shows the information required by all the business decisions identified for the Decision Points. It can be thought of as a decomposition of those business decisions into sub-decisions and areas of supporting information: business knowledge and data.

Figure 30 shows an example DRD.

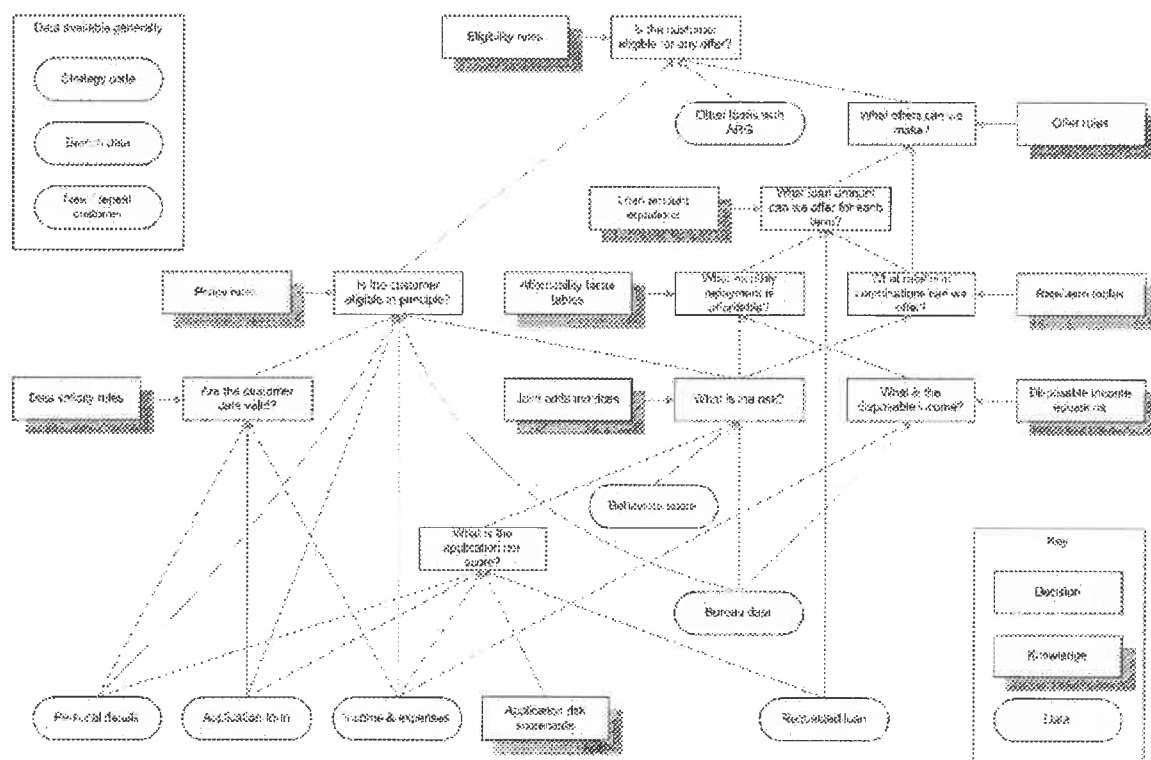


Figure 30—Sample Decision Requirements Diagram (DRD)

The DRD contains three types of nodes: decisions, knowledge and data. These are linked by arrows which indicate requirements: an arrow from A to B indicates that A is required for B. For example, in Figure 30 the decision “What is the risk?” requires four pieces of information: the result of the decision “What is the application risk score?” business knowledge expressed as joint odds matrices, the behavior score, and bureau data.

By convention decisions are decomposed downwards, so the arrows normally point upwards. Note that the diagram is not necessarily a tree: a single piece of information (e.g. the Income & Expenses data in Figure 30) may be required by multiple decisions.

The DRD is intended to give an “at a glance” view of the scope and structure of the required decisioning. The following sections of the DDD provide definitions of all the nodes on the diagram: decision nodes, knowledge nodes and data nodes.



4.7.2.3.2 Decisions

This section provides a definition of every decision node on the DRD, covering:

- *Name:* the name of the decision
- *Question:* a natural-language question characterizing the decision
- *Definition:* a simple definition of the decision, explaining how areas of knowledge and data are used to determine the result
- *Business Knowledge:* a list of all the areas of business knowledge used in the decision (knowledge nodes on the DRD)
- *Input Data:* a list of all the areas of data used in the decision (data nodes on the DRD)
- *Output Data:* a list of all the data resulting from the decision (including the decision value itself plus reasons or explanations for the decision).

Note that all business terms used to define the decisions must be documented in the Business Terms Catalog in the Decision Harvesting Worksheets.

4.7.2.3.3 Business Knowledge

This section provides a definition of every knowledge node on the DRD, covering:

- *Name:* the name of the knowledge area
- *Description:* a high-level description of the form of the business knowledge (rules, tables calculations etc) and its function
- *Example:* where possible, an example of the business knowledge (e.g. a couple of rules or a section of a table). Note that any rules, tables or equations provided here are for illustration only, to aid understanding, and do not form part of the requirement: the actual rules and calculations to be implemented will be collected and defined in the Knowledge Worksheets.
- *Knowledge Sources:* the sources of the knowledge (e.g. people or documents)
- *Maintenance Requirements:* who needs to maintain the knowledge, how, and how often,

4.7.2.3.4 Data

This section provides a definition of every data node on the DRD, covering:

- *Name:* the name of the data area
- *Description:* a simple high-level description of the data

4.7.2.4 Scope

This section defines the scope of the Blaze implementation by drawing a boundary around a subset of the nodes on the DRD, as shown in Figure 31, and describing verbally which nodes are in scope and which are not. This scope boundary will be used in subsequent tasks:

- All nodes defined as in scope must be accounted for in the design, which will specify how each node is to be implemented.
- All nodes defined as out of scope imply a data interface (wherever the requirements arrow crosses the scope boundary) which must be covered in the design and in the data model.
- The scope may be partitioned between a number of smaller boundaries for iteration planning.

4.8 Supporting Requirements Document

Examples of BRMS Supporting Requirements

- And much more.



4.8.1 Supporting Requirements Categories

Supporting requirements are categorized according to the FURPS+ model (Functionality, Usability, Reliability, Performance, Supportability, and +Constraints). +Constraints include design, implementation, interfaces, physical constraints, and business rules. A description of each of these types of requirements follows.

Supporting requirements and Use Cases, together, define the requirements of the system. These requirements support the features listed in the Vision statement. Each requirement should support at least one feature, and each feature should be supported by at least one requirement.

There is a finite set of requirements to consider when it comes to gathering system-wide requirements, qualities, or constraints. Many of them are unfamiliar to stakeholders; therefore, they may find it difficult to answer questions related to topics such as availability, performance, scalability, or localization. You can use this guideline when speaking to stakeholders to ensure that all topics are discussed. Make sure that stakeholders understand the costs of their selections and do not end up wanting everything that is on the list.

4.8.1.1 Functionality requirements

Functionality requirements include all the overarching, system wide functional requirements. These functional requirements represent the main system features that are familiar within the business domain or technically oriented requirements such as auditing, licensing, localization, mail, online help, printing, reporting, security, system management, or workflow.

Examples of Functionality requirements are:

- *Auditing:* Is there a need to track who used the system and when they used it? State requirements for providing audit trails when running the system.
- *Authentication:* Will access to the system be controlled? State requirements for authentication.
- *Licensing:* Will the system or parts of the system be licensed? If open-source software has been used in the system, have all open-source agreements been respected? State requirements for acquiring, installing, tracking, and monitoring licenses.
- *Printing:* Will printing capability be required? State requirements for printing.
- *Reporting:* Is reporting capability required? State requirements for reporting.
- *Scheduling:* Will certain system actions need to be scheduled? State requirements for scheduling capability.
- *Security:* Will elements of the system or system data need to be secure? State requirements to protect access to certain resources or information.

4.8.1.2 Usability requirements

Usability requirements include requirements based on human factors and user interface issues such as accessibility, interface aesthetics, and consistency within the user interface. These requirements are especially important for the Rule Maintenance Application (RMA).

Usability requirements are critical to the success of any system. Unfortunately, usability requirements are often the most poorly specified requirements. Consider this common requirement: The system shall be easy to use. This doesn't help much, because it cannot be verified.

While capturing usability requirements, it is a good idea to identify issues and concerns first, and then refine these into verifiable requirements later. According to a traditional definition, usability consists of five factors:

1. *Ease of learning:* A user with a specified level of experience must be able to learn how to use the system in a specified amount of time.



2. *Task efficiency:* A user should be able to complete a specified task in a specified time (or number of mouse clicks).
3. *Ease of remembering:* A user should be able to remember how to accomplish specified tasks after not using the system for a specified period of time.
4. *Understandability:* A user must be able to understand system prompts and messages and what the system does.
5. *Subjective satisfaction:* A specified percentage of the user community must express satisfaction with using the system.

You may want to use the following method to identify and specify usability requirements:

1. Identify the key usability issues by looking at critical tasks, user profiles, system goals, and previous usability problems.
2. Choose the appropriate style to express the requirements:
 - *Performance style:* Specify how fast users can learn various tasks and how fast they can perform the tasks after training.
 - *Defect style:* Rather than measuring task times, identify usability defects and specifies how frequently they may occur.
 - *Guideline style:* Specify the general appearance and response time of the user interface by reference to an accepted and well-defined standard
3. Write the actual requirements, including performance criteria

4.8.1.3 Reliability requirements

Reliability requirements include aspects such as availability, accuracy, predictability, frequency of failure or recoverability of the system from shut-down failure.

Reliability includes the system's ability to continue running under stress and adverse conditions. In the case of an application, reliability relates to the amount of time that the software is available and running as opposed to time unavailable. Specify reliability acceptance levels, as well as how they will be measured and evaluated. Describe reliability criteria in measurable terms. This is usually expressed as the allowable time between failures or the total allowable failure rate. Other important reliability considerations include:

- *Accuracy:* Specify requirements for the precision (resolution) and the accuracy (by some known standard) that is required in any calculation performed or in system output.
- *Availability:* Specify requirements for the percentage of time the system is available for use, hours of use, maintenance access, and degraded-mode operations. Availability is typically specified in terms of the Mean Time Between Failures (MTBF).
- *Recoverability:* Specify requirements for recovery from failure. This is typically specified in terms of the Mean Time to Repair (MTTR).
- *Frequency and severity of failures:* Specify the maximum defect rate (typically expressed as defects/KSLOC or defects/function-point) and severity of failures. Severity may be categorized in terms of minor, significant, and critical defects. The requirements must define each of these terms unambiguously. For example, a critical defect could be defined as one that results in loss of data or complete inability to use certain functionality of the system.

4.8.1.4 Performance requirements

Performance requirements address concerns such as throughput of information through the system, system response time and resource usage.



- **Response time:** Specify the amount of time available for the system to complete specified tasks and transactions (average, maximum). Use units of measurement.

Examples:

- Any interface between a user and the system shall have a maximum response time of 2 seconds.
- The product shall download the new status parameters within 5 minutes of a change.
- **Throughput:** Specify the capacity of the system to support a given flow of information (for example, transactions per second).
- **Capacity:** Specify on the volumes that the product must be able to deal with and the numbers of data stored by the product. Make sure that the requirement description is quantified, and thus can be tested. Use unit of measurement such as: the number of customers or transactions the system can accommodate, resource usage (memory, disk, ...) or degradation modes (what is the acceptable mode of operation when the system has been degraded in some manner)

Examples:

- The product shall cater for 300 simultaneous users within the period from 9:00 AM to 11 AM.
- Maximum loading at other periods will be 150.
- Start-up: The time for the system to start up.
- Shut-down: The time for the system to shut down

4.8.1.5 Supportability requirements

Supportability requirements include requirements such as compatibility and the abilities to test, adapt, maintain, configure, install, scale, localize, and so on.

Examples of supportability requirements include:

- **Adaptability:** Are there any special requirements regarding adaptation of the software (including upgrading)? List requirements for the ease with which the system is adapted to new environments.
- **Compatibility:** Are there any requirements regarding this system and its compatibility with previous versions of this system or legacy systems that provide the same capability?
- **Configurability:** Will the product be configured after it has been deployed? In what way will the system be configured?
- **Installation:** State any special requirements regarding system installation
- **Level of Support:** What is the level of support that the product requires? This is often done using a Help desk. If there are to be people who provide support for the product, is that support considered part of what you are providing? Are there any requirements for that support? You might also build support into the product itself, in which case this is the place to write those requirements. Consider the level of support that you anticipate providing and what forms it might take.
- **Maintainability:** Are there any special requirements regarding system maintenance? What are the requirements for the intended release cycle for the product and the form that the release will take? Quantify the time necessary to make specified changes to the product. There may also be special requirements for maintainability, such as a requirement that the product must be able to be maintained by its end-users or developers who are not your development team. This has an effect on the way that the product is developed, and there may be additional requirements



for documentation or training. Describe the type of maintenance and the amount of effort required.

Examples:

A new weather station must be able to be added to the system overnight.

The maintenance releases will be offered to end-users once a year.

- *Scalability*: What volumes of users and data will the system support? This specifies the expected increases in size that the product must be able to handle as businesses grow (or are expected to grow); the software products must increase their capacities to cope with the new volumes. This may be expressed as a profile over time.
- *Testability*: Are there any special requirements regarding the testability of the system?

4.8.1.6 + Constraints

The + of the FURPS+ acronym allows you to specify constraints, such as design, implementation, interfaces, physical constraints, and business rules:

- *Design constraints* limit the design and state requirements on the approach that should be taken in developing the system. E.g., are there any design decisions that have been mandated that the product must adhere to?
- *Implementation constraints* put limits on coding or construction (required standards, languages, tools, or platform)
- *Interface constraints* are requirements to interact with external systems, describing protocols or the nature of the information that is passed across that interface.
- *Physical constraints* affect the hardware or packaging housing the system (shape, size, and weight).
- *Business rules* are policies or decisions that govern how the business operates. They may constrain the steps described in the Use Case flow.

User Interface Requirements (+)

Describe both the user interface and interfaces with external systems. It is very important to document end-user supporting requirements for the Rule Maintenance Application because the perceived success of failure of maintainability/extensibility aspects of a BRMS will hinge upon the end-user comfort with and acceptance of the RMA.

Describe requirements related to user interfaces that are to be implemented by the software. The intention of this section is to state the requirements, but not to describe the user interface itself, because interface design may overlap the requirements-gathering process. This is particularly true if you are using prototyping as part of your requirements gathering process. As you develop prototypes, it is important to capture the requirements that relate to the look and feel of the user interface. In other words, be sure that you understand the business's intentions for the product's look and feel. Record these as requirements, rather than merely using a prototype for approval.

- *Look and feel*: A description of the aesthetic appearance and layout of the interface. Your end-user may have given you particular demands, such as style, colors, degree of interaction, and so on. This section captures the requirements for the interface, rather than the design for the interface. The motivation is to capture the expectations, the constraints, and the user's demands for the interface before designing it.
- *Layout and navigation requirements*: Specify requirements on major screen areas and how they should be grouped together.



- **Consistency:** Consistency in the user interface enables users to predict what will happen. This section states requirements on the use of mechanisms to be employed in the user interface. This applies both within the system, and with other systems and can be applied at different levels: navigation controls, screen areas sizes and shapes, placements for entering / presenting data, terminology
- **User personalization and customization requirements:** Requirements on content that should automatically displayed to users or available based on user attributes. Sometimes users allowed to customize the content displayed or to personalize displayed content.

Interfaces to external systems or devices (+)

- **Software interfaces:** Are there any external systems with which this system must interface? Are there any constraints on the nature of the interface between this system and any external system, such as the format of data passed between these systems? Do they use any particular protocol? Describe software interfaces with other components. These may be purchased components, components reused from another application, or components being developed for subsystems outside of the scope of the system under consideration, but with which this it must interact. For each system, consider both provided and required interfaces.
- **Hardware interfaces:** Define any hardware interfaces that are to be supported by the software, including logical structure, physical addresses, expected behavior, and so on.
- **Communications interfaces:** Describe any communications interfaces to other systems or devices, such as local area networks (LANs), remote serial devices, and so on.

4.9 Decision Harvesting (a.k.a., Rule Harvesting)

Decision Harvesting identifies all the business rules and computations for each decision and sub-decision to be made, documents every term used in every rule, identifies all the logical relationships between decisions, and develops a high level business object model. The results are recorded in the Decision Harvesting Workbook

In Blaze Implementation projects, Decision Harvesting is usually the most significant part of the process of Detailing Requirements. Decision Harvesting starts with the Outline Decision Requirements, which define as a minimum all the decision points and principal decisions, and exhaustively identifies all the business logic below that level:

- all the sub-decisions of the principal decisions
- every business rule and computation for each decision
- every term used in every rule
- all the logical relationships between decisions
- a high level business object model.

Decision Harvesting takes place in Decision Harvesting Sessions involving a FICO Rule Analyst and business Subject Matter Experts (SMEs). All the resulting information is recorded in the Decision Harvesting Workbook, which is signed-off by the business as the basis for design and development. This document is central to the subsequent implementation process: it is the sole definition of detailed decisioning requirements for the decision services to be implemented.

Decision Harvesting may be carried out in a single task, or may be split over a number of iterations, depending on the circumstances of the project. If iterative harvesting is used, each increment might correspond to (e.g.):

- One or more decision points



- One or more decisions
- A subset of knowledge nodes on a DRD
- One or more sources of business knowledge (rarely used).

If there are substantial numbers of rules involved, increments may be further sub-divided into tranches of rules.

4.9.1 Resources and Responsibilities

The following is an overview of the required resources and responsibilities. It is useful for planning decision harvesting workshops.

Who should be there?

- SMEs: People who have a detailed knowledge of the decision in the system under discussion and the rules associated with each decision (it is not uncommon for more than one person to be involved).
- The Rule Analyst (who facilitates the sessions) and a Scribe (who documents the results). These may be the same person but having two Rule Analysts works better.

What should the SMEs bring?

- Knowledge of the business process under discussion
- Supporting documentation, relevant
- List of those with decision making authority for decisions in this business process.

What the Decision Rule Analyst brings:

- A brief introduction of what you expect to accomplish and your expectations of the other participants.
- Some way to capture white board information (e.g. laptop with word processor, digital camera, scribe & notebook).

What kind of facilities is needed?

- White boards and flip charts, a projector is very useful. You'll need to be updating the Business Terms Catalog, Structured Rule Work Sheets and Decision Dependencies Worksheet.

4.9.2 Inputs

The following are typical inputs to the decision harvesting process:

4.9.2.1 The High-level Delivery Plan

This includes:

- the scope of the increments
- the increments assignment to this iteration.

4.9.2.2 The Outline Decision Requirements

These provide as a minimum:

- the decision points: those points in the business process where decision services are required to make decisions
- the principal decisions to be made at those decision points



If DRA has been used, the DDD will provide additional information, such as:

- the structure of the decisioning (e.g. dependencies between decisions; re-use of decisions)
- the areas of business knowledge to be harvested, with estimates of their size, complexity and maintenance requirements
- the areas of input data required by the decisioning.

This additional information is not required for the harvesting, which is a top-down process, but may be useful for guiding its activities, checking the results, and/or registering changes to the functional scope as agreed in the DDD.

4.9.3 Outputs

The following are typical outputs from the decision harvesting process:

4.9.3.1 The Decision Harvesting Workbook, including:

- Business Term Catalogue
- Decision Dependencies Worksheet *
- Rule Worksheets
- Abstractions Worksheets
- Computations Worksheets.

* The Decision Dependencies Worksheet is optional provided that:

- DRAW has been carried out by a highly experienced analyst and all decision dependencies have been captured in the DDD
- If during Decision Harvesting any new decisions or decision dependencies are discovered, these must be documented, either by including a Decision Dependencies Worksheet or by updating the DDD."

4.9.4 Preparation

The Rule Analyst should ensure that s/he is familiar with the domain by reading background on the business and the specific business process, collecting existing documents, and understanding the Outline Requirements.

4.9.5 Process Steps

During the Decision Harvesting Sessions rules are detailed and documented within the Structured and/or Generic Rules and/or Hybrid worksheets. Here are some initial steps to start to documenting the Decisions from the business perspective as defined in the Design Definition Document. These steps represent a set of activities that are done in parallel to iteratively define the detailed decision requirements.

4.9.6 Capture and Document the Rules

The following outlines the high-level process for capturing and documenting rules.

1. Work top-down beginning with the highest decision in scope.
2. Document each decision and its decision point in the Decision Dependencies Worksheet
Determine if each decision is derived by rules or a computation and document it using the appropriate worksheet.



If the decision is derived by use of a computation document it in the Computation Worksheet.

If the decision is being derived by use of rules it should be documented in a Rules Worksheet with the decision name used as the name on the worksheet tab.

- If a decision is derived by use of several rules but the rules do not share the same conditions. Use the Structured Rules Worksheet would leave the majority of cells blank and be cumbersome to complete. In this case the Generic Rules Worksheet would be the best choice.
- If a decision is derived by use of several rules and all the rule have a few conditions in common e.g. claim type and provider type but the remaining conditions are almost always different the Hybrid Rule Worksheet would be the best choice.
- In the majority of cases a Structured Rules Worksheet should be used, this structure of the Rules Worksheet allows for the greatest flexibility and ease of use in Decision and Rule Analysis.

When documenting the business rules make sure:

- ☒ Each individual rule passes its individual rule quality check and can be documented in the appropriate worksheet
- ☒ Each rule in a Rules Worksheet must have the same action (decision) as other rules in that Rule Worksheet.
- ☒ Each business term that is a condition on the left hand side of the rule (the "if" side) is used as a column header in the conditions section if a structured rule worksheet is used. The business term to be derived, the decision, is placed as a column header in the action section of the spreadsheet (there may be more than one action).
- ☒ Each rule (requirement) is identified by a unique decision/rule-based ID. This is a combination of the decision-name and the rule identifier. For example: ProcessLoan_R01 for the first rule in the decision Process Loan. This is referenced by the Test Cases to identify which specific decision-based requirements are tested by the Test Case. Further guidance on traceability can be found at: Guidelines: Traceability of Blaze Decision Requirements.

If the decision is derived by rules a copy of the "Structured rules Worksheet" should be made and its tab name should be the name of the decision. If the derivation of the business term is by means of a computation it should be documented in the "Computation worksheet" tab.

3. Validate the rule

Each a rule needs to pass some preliminary rule analysis:

- ☒ Each rule should express one complete thought (i.e. it cannot be further decomposed). There may be some exceptions to this in BRE optimization considerations but for decision harvesting this should be enforced.
- ☒ All the conditions on the Left Hand Side (LHS) of the rule must contain all and only the necessary conditions for that rule.
- ☒ Each rule should be documented in an implementation independent format.
- ☒ Establish individual rule quality (Atomic and Concise).

4.9.7 Capture and Document the Business Terms associated with each Rule

Every business term in each rule and/or computation needs to be documented in the Business Terms Catalog. This can be completed in conjunction with documenting the business rules in the Rules Worksheet.



The Business Term Catalog contains valuable information about each business term and is the basis for creating the Blaze Business Object Model.

Documentation of the business terms must follow the following procedure:

Business Term	Definition	of a	Source of Term Value	Term Type	Location of Derivation	Valid Values	Default Value	Other Meta-Data	Business Rule
Customer	An individual or business that purchases products		Input	Entity					
Order	Contains information about what the customer has purchased	Customer	Input	Entity					
Date of Birth	The date the customer was born	Customer	Input	Input				DOB	
Age	How old the customer is at the present time	Customer	Derived	Transfers	Computation Worksheet	BETWEEN 0 AND 123			
Percent Discount	The percentage that will be deducted from the Order Amount	Order	Derived	Output		BETWEEN 0 AND 100%	0		Discount
Order Amount	Dollar value for the order before discounts, taxes, fees and shipping	Order	Derived	Output			0		
Shipping Cost	Dollar amount that will be charged to ship the order to the customer	Order	Derived	Output			0		
Business Rule	A statement that defines or constrains some aspect of the business. This may be a term or fact described below as a structural attribute of a concept, identified below as an action, condition, or a definition. It is "logical" in that it cannot be broken			Concept Definition					
Derived Term	A term whose value is determined by the use of rules. It is the result of the right-hand side of a rule. It is a term whose value is determined during the decision processing. The term value may be stored in a database, the subject etc.			Concept Definition					

Figure 32—Sample Business Term Catalog

1. Enter term into the Business Term Catalog
2. If the definition of this term is known it should be entered in the "Definition" cell.
3. If the Entity to which this term belongs is known (or reasonably guessed) it should be entered in the "of a" cell (e.g. driving risk of a customer, total amount of an order).
4. For Business Terms that are a result of a decision the "Source of Term Value" cell should be set to "Derived".
5. If the valid values for this Business Term are known they should be entered in the "Valid Values" cell. Valid Values can be helpful in rules harvesting as they can help direct the search for missing or hidden rules for that decision (i.e. there should be at least one rule for each of the valid values).
6. If there is a default value for the decision it should be entered in the "Default Value" cell (e.g. the default value for total amount of an order may be zero).
7. The other meta-data items can be ignored at this point and completed later.
8. If the Business Term Type is a Concept Definition then only the business term and its definition need to be entered.
9. If the Business Term Type is Entity then the business term and its definition must be entered and other meta-data is a nice to have for example valid values.
10. If the Business Term Type is Input then the business term, definition, of a, source of term value, and term type, are required. Source of Term Value and Term Type are set to "Input".
11. If the Business Term Type is Output then the business term, definition, of a, source of term value (set to derived), term type (set to output) and Location of derivation are required. Also if Valid Values and Default Value are known it will be of value to enter them at this time. If the business term is a condition, on the left hand side of a rule, it must be documented in the Decision Dependencies Worksheet.



12. If the Business Term Type is Transient then the business term, definition, of a, source of term value (set to derived), term type (set to transient) and Location of derivation are required. Also if Valid Values and Default Value are know it will be of value.
13. It is not uncommon to have several terms in the Business Term Catalog that do not have a definition or are missing metadata. The business rules analyst will need to set up sessions with SMEs and other business folks to resolve Business Term Catalog issues.

4.9.7.1 Example:

In this example Shipping Cost is the decision to be derived. Every rule used to derive Shipping cost needs to be identified. As rules are added to the worksheet each business term that is a condition used to help derive the action/decision on the right hand side of the rule is entered as a column header. The same holds for the action/decision.

Each row of the spreadsheet can be read as a rule for example the first row would be read as "If age greater than or equal to 18 and order amount greater than or equal to \$100 and percent discount less than 10% then shipping cost equal \$4.99". At this point each rule has gone through some quality control for individual rule quality but nothing at the decision level. As the work sheet is filled out analysis will need to be done.

Conditions					Then		Action/Decision				
Age	Order Amount	Percent Discount	Condition Attribute	Condition Attribute	Shipping Cost	Condition Attribute	Rule Identifier	Condition	Business Type	Rule Identifier	Comments
>= 18	>= \$100	< 10%			\$4.99		ShippingCostRule1	Age >= 18	Transient	ShippingCostRule1	Shipping cost is derived from Age
< 18	>= \$100	< 10%			\$6.99		ShippingCostRule2	Age < 18	Transient	ShippingCostRule2	Shipping cost is derived from Age
>= 18	>= \$100	< 10%			\$4.99		ShippingCostRule3	Age >= 18	Transient	ShippingCostRule3	Shipping cost is derived from Age

Figure 33—Sample Decision Requirement Worksheet

4.9.8 Capture and Document Decision Dependencies

The initial entry in the Decision Dependency worksheet is from the DDD. A new row should be added to the Decision Dependencies Worksheet. The left most two entries should be the name of the Business Term and name of the Decision Point. Decision Term refers to the business term whose value is being derived at that decision point. At this step in the harvesting process one is identifying decisions and decision points not dependences consequently there is no need to fill in the right hand side of the Decision Dependencies Worksheet.

If a condition of a rule being documented in the Business Term Catalog has a Source of Term Value equal to "Derived" then a new row should be added to the Decision Dependencies Worksheet (if one does not exist). The business term should be entered on the left hand side of the Decision Dependencies Worksheet under Business Term and the business term that is used in the Action portion of the rule should be entered on the right hand side of the Decision Dependencies Worksheet under Business Term. The information being captured here is that the decision trying to be derived with this rule cannot be determined until the value of the business term is determined and there is a dependency.

The Decision Dependencies Worksheet is optional provided that:

- If during Decision Harvesting any new decisions or decision dependencies are discovered, these must be documented, either by including a Decision Dependencies Worksheet or by updating the DDD.*



Business Term	Decision Point	Iteration	Logically Depends on	Decision Point	Business Term	Iteration
Customer Status	Determine Customer Status		Logically Depends on	Determine Customer Eligibility	Customer Eligibility	
Application Status	Decide to Approve or Reject Application		Logically Depends on	Determine Customer Status	Customer Status	
Interest Rate	Determine Interest Rate on Loan		Logically Depends on	Decide to Approve or Reject Application	Application Status	

Figure 34—Sample Decision Dependencies Worksheet

4.9.9 Capture and Document Abstractions

In some cases the rules have similar patterns of multiple tests in common. Where this is often repeated, it will be useful to abstract this concept into something higher level. A form of abstraction can also be used at the term level. There may be several terms that naturally form a group (e.g. RI, CT, MA, VT, NH etc. compose NEW ENGLAND).

If a group of terms are conceptually related then use of an abstraction may be very helpful in limiting the number of conditions, rules, complexity of the rules and help in make the rules more intuitive.

Abstraction will also help rule maintenance as the abstraction changes the only rule that needs to be modified is the rule that creates the abstraction.

All abstractions should be documented in the Business Term Catalog. A best practice is to use abstraction only the rules or terms form a natural grouping, the conditions or terms are conceptually related.

Example: Common multiple condition abstraction

```

IF
    Family Code is NC
    AND Occupancy is Primary
    AND Property Type is Single Unit
    AND Building Status is Existing
    AND Property Class is not Manufactured
    AND Program Code is Time Saver Refinance Extra

THEN
    Appraisal Product = medium level
  
```

Rules with Abstraction Added:

```

IF
    Family Code is NC
    AND Occupancy is Primary
    AND Property Type is Single Unit
    AND Building Status is Existing
    AND Property Class is not Manufactured

THEN
    Non-Conforming Simple Home = True
  
```



IF
 Non-Conforming Simple Home is True
 AND Program Code is Time Saver Refinance Extra

THEN
 Appraisal Product = medium level

This is useful when the same group of conditions occur in many rule and the conditions are conceptually related i.e. the new abstraction is intuitive. If any of the conditions to determine Non-Conforming Simple Home change they only need to be changed in the one rule that determines that condition not the many that use the abstraction.

To document abstraction make a copy of the "Generic Rule worksheet" you may choose to use a general one for all abstractions labeled abstraction or for more sophisticated ones you may choose to label the tab with the name of the abstraction. As always, all terms need to be documented in the Business Terms Catalog.

4.9.10 Analyze Rules for Consistency, Completeness and Redundancies

Once the worksheets have been completed within the Decision harvesting sessions and before they are sent out for approval, the team should go back through them to ensure they are of good quality. Here is a checklist of things to consider:

- ☒ All rules for each decision have been discovered (completeness)
- ☒ Should abstraction and/or enumerations be applied
- ☒ Remove redundant rules
- ☒ Compare logic among rules that have the same then condition based on the same "if conditions"; remove duplicates
- ☒ Resolve overlaps among rules
- ☒ Compare "if conditions" across rules with the same "then condition" and check for overlapping conditions
- ☒ Resolve Inconsistencies among rules; it should never be possible to conclude incompatible results give the same information and decision set
- ☒ Compare rule clauses in the same decision set for equivalent conditions but inconsistent "then conditions"
- ☒ Compare rule clauses in the same decision set for inconsistent "if conditions" that infer equivalent "then conditions"
- ☒ Ensure completeness of decision sets
- ☒ Check for module completeness by examining all possibilities for the "if conditions i.e. sort by gender, age, amount etc.
- ☒ Make sure a rule is not subsumed by another
- ☒ Check the Business Terms Catalog for missing items. This should be done frequently. A good practice is to give homework assignments between meetings with assignments to team members to help fill in missing items. Usually team members are willing to volunteer with a little prodding.

4.9.11 Review and Obtain Approval/Sign-off on Rules

Schedule a preliminary review meeting to review the requirements of the given scope with the SME's, other noted stakeholders. Make adjustments to the worksheets as necessary. Analysis at the Meta-data and decision dependency level should be reevaluated.

Once all changes have been made, schedule a final review meeting with, at a minimum, those who will approve the requirements. An attempt should be made to have as many, if not all, of the deliverables signed off before the meeting. If the documents are not signed off, at the very least, a verbal approval should be given by the SME. There should be no surprises at this final review meeting.

4.10 Decision Harvesting Workbooks

The Business Terms Catalog is the map between Decision Harvesting entities

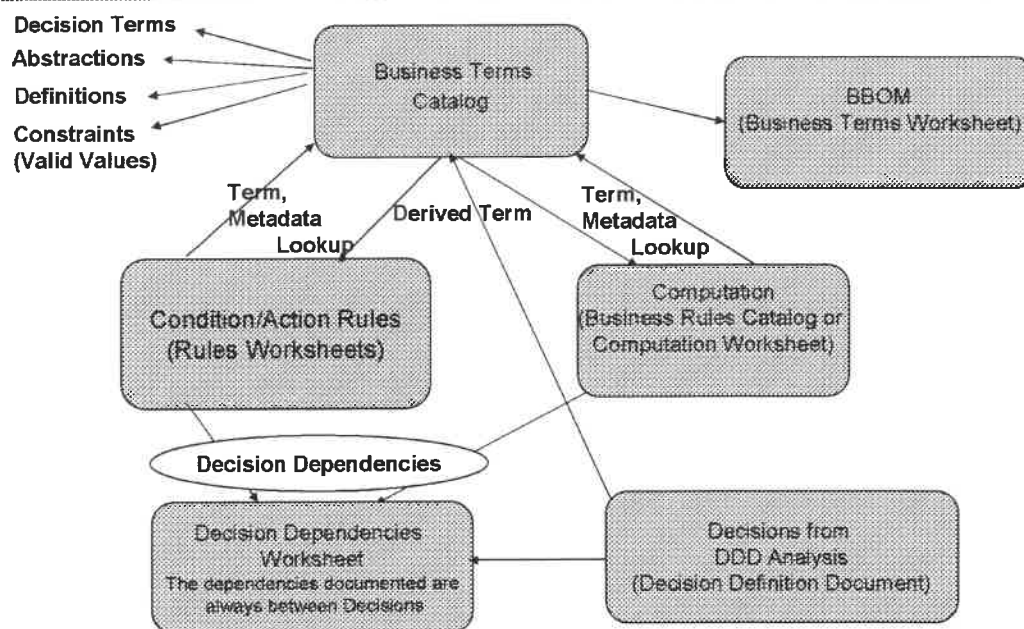


Figure 38—Organization of a Decision Harvesting Workbook

4.10.1 Purpose and Scope

Decision Harvesting Worksheets are used to capture all of the business rules associated with a business process.

4.10.2 Business Term Catalog

Each entry (business term) in the Business Term Catalog has some specific meaning to the business community and is used by the business rules. This includes the data this is used to make the decision (conditions) and the decision itself (actions). Each Business Term has a definition and other pertinent meta-data the helps define the term.



For Example:

- The Term Type is identified for each business term e.g. a concept definition, entity, input to a decision, output of a decision, both input and output or transient (used in decision making but neither input nor output).
- If the term is used in a rule the source of the term is identified as either input or derived.
- If the term is derived, then the location of its derivation is identified.

Additional meta-data is captured for each term as needed depending on the needs of the engagement.

A high level Business Object Model is the result of properly completing the Business Term Catalog. The Business Term Catalog also serves as a map between Decision Harvesting work products.

The definition of each column is described in the "Rule Worksheet Column Def" worksheet of the Business Term Catalog and Decision Rule Worksheet spreadsheet.

4.10.3 Decision Dependencies Worksheet

The Decision Dependencies Worksheet's primary purpose is to identify logical dependencies between decisions. Some decisions depend on the outcome of other decisions before they can be made. In the process of accomplishing this goal there are two methods for decisions to be entered in this worksheet:

1. A decision point is discovered that is in the scope of this application. In this case the name of the decision point and the name of the decision associated with the decision point are entered on the first two columns on the left hand side of the worksheet.
2. If while documenting a business term in the Business Term Catalog it is determined that the "Source of term value" is "derived", that term should be placed in the Decision Dependencies Worksheet in the right most cell under Business term. In the same row the term on the action side of the rule (the then side) should be placed in the left hand most cell of that row under business term. In this case "Decision Point" need not be filled in.

This worksheet is input to design and shows which decisions need to be made before others can be made. It is also useful for impact analysis as it will show how change to one decision will impact other decisions.

4.10.4 Generic Rule Worksheet

The Generic Rule Worksheet is employed in unique cases when it is not possible to use the Structured Rules Worksheet to document business rules. Each rule has a unique identifier assigned to it as well as other meta-data such as owner, source type, rule analyst start date, stop date, etc.

The amount of meta-data captured is determined by the needs of the business. The meta-data options will be the same for the Structured Rule Worksheet. This worksheet is for special cases and will be used much less frequently than the Structured Rule worksheet.

This worksheet is also the "template" that is used for the Computation Worksheet and the Abstraction Worksheet.

The definition of each column is described in the "Rule Worksheet Column Def" worksheet of the Business Term Catalog and Decision Rule Worksheet spreadsheet.

4.10.5 Structured Rule Worksheet

The Structured Rule Worksheet is used to document all of the business rules used to derive a particular decision. The rules are documented in a simple "If Then" format. There can be several conditions in the "If" portion of the rule and one or more actions on the "Then" portion of the rule.



This worksheet has a format similar to a decision table. Each business rule has a unique identifier as well as other meta-data such as owner, source type, rule analyst, start date, stop date, etc. The amount of meta-data captured is determined by the needs of the business.

The vast majority of decisions will have their rules documented using this worksheet.

The definition of each column is described in the "Rule Worksheet Column Def" worksheet of the Business Term Catalog and Decision Rule Worksheet spreadsheet.

4.10.6 Computation Worksheet

The Computation Worksheet is used to document all computations identified during the Decision/Rule Harvesting process. This worksheet uses the Generic Rule worksheet as its foundation. It consists of two main columns; one that identifies the Business Term and a second that contains the computation (or calculation) that defines the Business Term. There is also a column for parenthetical remarks. As with other rules a unique identifier assigned to the computation as well as other meta-data such as owner, source type, start date, stop date, etc. The amount of meta-data captured is determined by the needs of the business. All computations for a business process should be documented in this worksheet.

4.10.7 Abstraction Worksheet

In some cases rules have similar patterns of multiple tests in common. Where this is often repeated, it will be useful to abstract this concept into something at a higher level. A form of abstraction can also be used at the term level. There may be several terms that naturally form a group. For example, RI, CT, MA, VT, and NH compose NEW ENGLAND. Thus, NEW ENGLAND would be the abstraction.

If a group of terms are conceptually related then use of an abstraction may be very useful in limiting the number of rules, the complexity of the rules and help make the rules more intuitive.

The Abstraction Worksheet uses the Generic Rule Worksheet as its foundation. Each abstraction will have a unique identifier assigned to it as well as other meta-data such as owner, source type, start date, stop date, etc. The amount of meta-data captured is determined by the needs of the business.

The definition of each column is described in the "Rule Worksheet Column Def" worksheet of the Business Term Catalog and Decision Rule Worksheet spreadsheet.

4.11 Ancillary Rule Harvesting Activities and Artifacts

The business analysis, requirements, analysis and design disciplines within the FIRUP methodology are all highly coupled and interdependent. Further, the methodology is iterative in nature. As a result, there are many aspects of rule harvesting that are intertwined with other disciplines.

It is not uncommon for analysts to get caught up in the details of rule harvesting and overlook opportunities to enhance or further the elaboration upon many supporting requirements. Nonetheless, because the rule harvesting effort involves close interaction with business subject-matter experts and end-users, it is an excellent source for supporting requirements. The best and most experienced decision analysts are always on the watch for additional supporting requirements that can be documented and passed along to other disciplines or team on the project.

The rule harvesting effort will often stimulate business SME and end-users to elaborate on the following supporting requirements while discussing decisioning requirements.

- Unit testing requirements
- User acceptance testing requirements
- Rule maintenance and extension requirements



- RMA user requirements
- Roles and responsibilities
- Release and approval processes
- And, much more...

The best decision analysts will not simply document these requirements when they surface in DRA discussions. The best decision analysts will solicit and explore these requirements at every opportunity during the DRA discussions.

4.11.1 Unit Test Requirements Definition

The definition of unit testing requirements is the responsibility of the test director and not the decision analyst. Nonetheless, the decision analyst is in a very advantageous position for soliciting a wealth of unit testing requirements from the business SME and end users.

As each decisioning requirement is explored during the normal process of rule harvesting, the decision analyst should establish a rhythm of elaboration that includes exploring how the requirement should be tested. This information should then be documented in the Supporting Requirements Document or passed along to the Test Director. Many decision analysts will extend their rule harvesting worksheets and incorporate this information as part of their normal harvesting process.

Important information that a decision analyst can capture as part of decision requirements harvesting includes:

- What conditions will cause the rule to fire
- What conditions will cause the rule to fail
- What are the boundaries of allowable values
- Are there any unexpected values
- What values may cause an exception condition
- What are the expected results when the rule fires
- What are the expected results when the rule fails
- What is the expected response to an exception condition

4.11.2 User Acceptance Test (UAT) Requirements Definition

Similar to unit testing, the definition of UAT requirements is also the responsibility of the test director and not the decision analyst. Nonetheless, the decision analyst is also in a very advantageous position for gleaming a wealth of UAT requirements from the business SME and end users.

However, unlike unit testing, UAT requirements are not something the decision analyst needs to actively solicit. UAT requirements are something that the decision analyst simply needs to be ready to handle when a SME or end user elaborates upon during the course of normal decision analysis.

Some clues to UAT requirements can be captured as part of decision requirements harvesting include:

- Comments about expected outcomes
- Comments about unacceptable outcomes
- Comments about expected on-screen information
- Comments about expected content in system messages
- Comments about tolerance for interface response times



- Elaborations about things that were unfavorable about the old system or process that they look forward to being corrected in the new system
- Basically, anything the end-user or SME might say about the things they will consider in their acceptance of the system.

4.11.3 Rule Maintenance and Extension Requirements

By far, the biggest value propositions from using BRMS technologies precipitate from the ease of maintenance and extension. To maximize those benefits, the business SME and end-users need to convey to the rule architect as much information as is possible about how decisioning requirements are expected to change over time. The decision analyst can assist in this effort.

As each decisioning requirement is explored during the normal process of rule harvesting, the decision analyst should establish a rhythm of elaboration that includes exploring how the requirement will need to be maintained over time. This information should then be documented in the Supporting Requirements Document or passed along to the rule architect. Many decision analysts will extend their rule harvesting worksheets and incorporate this information as part of their normal harvesting process.

Important information that a decision analyst can capture as part of decision requirements harvesting includes:

- How is the requirement expected to change over time
- What aspects of the requirement are foreseen to change and what aspects are expected to be static
- Is there a need to copy/paste or create similar new rules
- Are there effective dates

4.11.4 RMA Requirements

In addition to rule maintenance and extension requirements, the decision analyst also needs to be on the lookout for, and even solicit end-user requirements for the rule maintenance application. These include

- User Requirements
- User Roles and Responsibilities
- Release Approval Process

This information should then be documented in the Supporting Requirements Document or passed along to the rule architect. Many decision analysts will extend their rule harvesting worksheets and incorporate this information as part of their normal harvesting process.

4.11.4.1 RMA User Requirements

RMA user requirements that SME and end-users often bring up during decision analysis sessions include the following:

- Corporate standards for look and feel (i.e., CSS)
- Preference for certain metaphors (i.e., Decision Tables, Trees, Scorecards, etc.)
- Requirements that are contextually related and need to be displayed together
- Semantics for business terms (i.e., "the address of the client" vs. "the client's address" vs. "theClient.Address")
- Usability features (i.e., editing accelerator/hot keys, undo/redo, rollback, split pane, etc.)



- Audit logs, revision history, version management tools, etc.

4.11.4.2 Rule Maintenance Roles and Responsibilities

The default user authentication and authorization (A&A) manager for the Blaze Advisor RMA provides only one user role and that role has authority to access and changes everything in the repository. Most of the time, the business has requirements to limit authority to certain aspects of the repository. If the decision analyst discovers situations where some areas of the requirements need to have access restrictions, or be read-only, these requirements should be documented in the Supporting Requirements Document (SRD) and passed along to the rule architect.

Typical authorities include:

- No access
- Read-only
- Author
- Review/Approve

These can be set at a global level, by subject-matter area, or on a requirement-by requirement basis.

4.11.4.3 Release Approval Process

While the BRMS methodology allows the business to maintain and extend business decisioning with minimal involvement from IT, it does NOT condone an absence of governance and rigor around the approval of modifications prior to release. The SRD should capture all non-functional requirements for the approval and release of modifications. These include:

- What roles are allowed to approve changes
- What levels of approval are required before a change can be released into each individual environment or lifecycle stage (DEV, TEST, Q/A, STAGE, PROD, etc.)
- What meta-data describes this approval process

4.11.5 Reporting, Logging, Audit, and Traceability Requirements

Similar to UAT requirements, gathering requirements for reporting, logging, audit, traceability, and similar requirement is not the responsibility of the decision analyst. Nonetheless, the decision analyst is also in a very advantageous position for gleaning a wealth of reporting requirements from the business SME and end users.

Similar to UAT requirements, reporting requirements are not something the decision analyst needs to actively solicit. Reporting requirements are something that the decision analyst simply needs to be ready to handle when a SME or end user elaborates upon during the course of normal decision analysis.

Some clues to reporting requirements can be captured as part of decision requirements harvesting include:

- | | |
|---|---|
| • Needs to show what is or is not in the repository | • Needs to show which rules were in effect when a decision was made |
| • Needs to show which rules fired or failed to fire | • Accountability to auditors |
| • Needs to show why rules fired or failed to fire | • Regulatory compliance |
| | • Burdon of proof |



5 Rule Mining Recommendations for CPI Print and Underwriting

Rule mining is a metaphorical term for the discovery of business rule or decisioning requirements from an existing codebase. It is used when an existing codebase is both the source of knowledge and the system of record for business rule or decisioning requirements.

In addition to the overview of FICO's FIRUP Methodology as it applies to the analysis and definition of decisioning requirements, Chubb has requested additional recommendations on a rule mining for the Chubb Personal Lines (CPI) Print and Underwriting projects.

This request is somewhat precarious for FICO because:

1. FICO's FIRUP methodology does not encourage rule mining because it is not an effective approach to harvesting and defining decisioning requirements. Under normal circumstances FICO would recommend Chubb NOT pursue a rule mining approach.
2. FICO wants to help Chubb achieve success on every application of Blaze Advisor BRMS within their enterprise. Chubb faces a challenge where the CPI Print and Underwriting projects do not have access to business SMEs and are therefore unable to engage in a FIRUP-like approach to decision analysis. Chubb may be in a situation where they have no choice other than to engage in a rule mining exercise.

This section is, therefore, included in this document for the purpose of providing Chubb with guidance in rule mining for particular cases where SME are not available. This section should in no way be construed as a recommendation for applying rule mining techniques to situations where SME or other authorities are available.

The subject-matter in this section includes:

- Overview of Rule Mining
- Business Rule Mining vs. Association Rule Mining
- Business Rule Mining Process Steps
- Rule Mining Recommendations for Chubb
 - Some Rules of Thumb about Rule Mining

5.1.1 Manual Rule-Mining Recommendations

The documents and workbooks that FICO uses for rule mining are precisely the same as those used for rule harvesting. FICO has provided Chubb with templates for these artifacts and they are available through Chubb's BRMS COE.

Recommendations include:

1. Allow the taxonomy for the workbooks to evolve as rules are mined.
2. Allow the DRD to evolve through iterative elaboration as rules are mined
3. Favor a hybrid strategy as opposed to strict top-down or bottom-up.
 - Subject-matter Experts (SME)
 - Subject-matter Expertise Is Needed for Rule Mining



5.2 Overview of Rule Mining

Today, as certain business software systems are approaching 20, 30, even 40 years of age; it is not uncommon to find situations where the business has adapted itself to existing software systems. Subject-matter experts may no longer be directly involved in day-to-day business decisions. In other cases, SMEs may not have a level of historical or detailed knowledge about all the decisioning that has accumulated in the system code over the years. In some cases, subject-matter experts may no longer even be available to the business. In all these situations, it is acceptable to undertake a "rule mining" approach to rule harvesting.

5.2.1 Business Rule Mining vs. Association Rule Mining

The discussions in this section are about Business Rule mining and are not to be confused with Association Rule Mining as such applies to data mining. In data mining, association rule learning is a popular and well researched method for discovering interesting relations between variables in large databases.

The classic example of association rule mining is an analysis of an enterprise-scale POS transaction database for a large supermarket chain. Analysis of the point-of-sale data might produce the rule:

{onions, potatoes} → {hamburger}

This association rule states that if a customer's order contains both onions and potatoes, he or she is likely to also buy hamburger.

While association rule mining is interesting and is often used to mine business rules from large sets of real-world data, this is not the subject-matter for this section on rule mining.

Business rule mining is the process of extracting decisioning logic in the form of Business Rule Requirements from legacy software applications, recasting them in natural or formal language, and storing them in a source rule repository for further analysis or forward engineering. The goal is to capture specifications for decisioning requirements so that the business can validate them.

Rule mining supports a business rules methodology but does not replace the formal analysis and definition processes for managing and automating an organization's business rules. Below is a general approach towards the semantic capture of technical rules and their abstraction into business rules.

The following subsections outline the high-level process for mining rule requirements from legacy software applications. For the most part, this process is equally applicable to both manual and automated approaches. The steps in the process are recommended to help build the right organizational framework and to align decision requirement mining with business priorities.

1. Avoid long-drawn-out rules mining exercises
2. Ensure that rules are valuable to the business
3. Speed the process of identifying business rules and not simply technical rules

5.3 Business Rule Mining Process Steps

The overall mining process is:

1. Define Project Goals and Desired Outputs
2. Select Rule Mining Technology
3. Define Project Approach and Time/Resources Allocation
4. Define Rule Mining Workflow



5. Determine Business Processes Mapping and Alignment
6. Perform Application Analysis
7. Define Business Term Catalog
8. Define Rule Taxonomy
9. Mine Business Rule Requirements (a.k.a., Candidate Rules)
10. Verify and Validate Candidate Rules
11. Transform Business Rule Requirements
12. Reports Production
13. Integration with a Target Environment
14. Proactively Manage Rules

5.3.1 Define Project Goals and Desired Outputs

It is important to define the goals and outputs of the rules mining activity. The goals for the effort will vary depending on business priorities. Depending on the goal there will be two primary outcomes that can be derived from a rules mining activity:

1. *Documentation*—in some cases, the goal may be to develop business-centric documentation of the business decisioning that is implemented within the program code segments. This documentation could be valuable as reference for other analysts and subject matter experts. It may also be used to feed high-level design efforts for modernization projects. It would not normally result in executable business rules or inputs into development environments.
2. *Business Rule Requirements*—in other cases, the exact semantic behavior of edits and calculations in applications needs to be captured as true business rule requirements. The semantics captured during rule mining will be used as a basis for business rule/decision requirements analysis and eventually business rule definition. The outcome of this step will drive decisions regarding technology adoption, resource planning, and the best methods to apply for either type of rule mining.

This first step is important because it defines the scope of the effort and limits the investment of time and resources by not producing work product associated with the desired outputs.

5.3.2 Select Rule Mining Technology

There are roughly 4 choices in selecting rule mining technology. Many tools are hybrids, but all tend to fall into one of the following 4 categories:

1. Manual Rule Mining Methodologies
2. Runtime Simulation Tools
3. Source Code Scanning Tools
4. Repository-Based Rule Mining Tools

Rule mining technologies, tools, and methodologies may also offer rule management and auditing capabilities. These are valuable for facilitating project workflow and rule maintenance activities by reviewers. These features are also valuable to retain rule traceability to originating sources even when those undergo change.



5.3.2.1 Manual Rule Mining Methodologies

By and far, the most common approach to mining rules is to follow a manual rule mining methodology. These methodologies offer reasonable value at a very low cost. The application of technology is generally limited to documentation of rules in word processing or spreadsheet applications.

The advantages to manual rule mining methodologies are:

1. Low up-front investment costs (utilizes COTS business office software)
2. Generally higher quality in work-product (due to human involvement in every step of the mining process)

The disadvantages to manual rule mining are:

1. Labor intensive
2. Duration may be protracted
3. Requires technical resources (to locate rules within source code segments)
4. Not an effective way to analyze application behavior

Manual approaches are typically recommended when:

- The scope is limited to one or two projects, or multiple small projects, where the investment costs for a tool is not offset by the savings in labor (typically, 48-to-60 man-weeks)
- The legacy applications are exceptionally monolithic and human involvement is needed at every step to decouple and abstract business concepts.

FICO generally recommends and follows manual methodologies when assisting its customers with rule mining. See Section 5.4, Rule Mining Recommendations for Chubb, for specific recommendations regarding this approach.

5.3.2.2 Runtime Simulation Tools

Runtime simulation tools are technologies that facilitate the capture of user behavior into processes and rules. These tools essentially simulate system inputs and process outputs in order to deduce behavior-oriented rules about the legacy application.

These tools are most helpful when the behavioral aspect of the mining is intensive and contraindicates a manual approach. The classic example for these types of tools is the analysis of a calculation utility such as a rating engine.

Their main drawback, however, is that they are limited to those test scenarios actually performed within a given time frame – potentially missing out on critical exceptions not usually performed. They are also not well-suited for large monolithic applications that take a broad spectrum of inputs, produce a broad spectrum of outputs, or perform a broad spectrum of functionality.

5.3.2.3 Source Code Scanning Tools

Source code scanning tools are text scanning tools that locate patterns within source files. These tools can accelerate the capture of rules based upon fixed patterns within source code.

These tools are rarely sufficient on their own and are typically used in conjunction with other tools or manual processes. These tools are very valuable when the legacy application has a large volume of source code and little documentation. The main draw-backs with these tools are:

1. They typically generate a large number of false positives, .e.g.,
 - a. Simple conditional execution logic is often misidentified as a business rule



- b. Format validations for things like dates and money are often misidentified as business rules
- c. Validations on ontology and data integrity are often misidentified as business rules

2. They tend to lead to technical rather than business rules

For these two reasons, FICO typically recommends source code scanning tools only to be used in conjunction with a manual process. This will allow a human analyst the opportunity to thin out the false positives and to aggregate the technical rules into business rule requirements.

5.3.2.4 Repository-Based Rule Mining Tools

There are new technologies and tools emerging in the market that recompile application sources, build a syntactic parse tree, and do a reasonably good job at identifying decisioning logic. The resulting structures and rules are stored in a repository as a system of record for the application's decisioning logic.

These repository-based technologies can offer the highest value in rule mining tools market. Some are very advanced and automatically detect rules using semantic analysis. However, expectations need to be managed. These tools are not cognitive and follow precise algorithms (e.g. each statement upstream from a point of interest variable and which potentially impacts its value is mined as a candidate rule). Similar to all source-code scanning technologies, these tools tend to produce a large number of false positive and tend to lead to technical rather than business rules.

For these reasons, FICO typically recommends automated semantic analysis tools only to be used in conjunction with a manual process. This will allow a human analyst the opportunity to thin out the false positives and to aggregate the technical rules into business rule requirements.

5.3.3 Define Project Approach and Time/Resources Allocation

The definition of project goals and selection of rule mining technology/methodology will impact the project approach and time/resource allocation. All too often, expectations from the business side are to receive a precisely modeled set of business rules derived from the application semantics, without realizing the complexity of such an effort. On the IT side, practitioners without experience in rules mining or familiarity with the application subject matter are ill-equipped to offer reliable estimates.

An iterative approach toward time and resource estimation may be adopted. Mining rules for a representative sub-set of the application over the first few weeks provides insights into the methods that work best for selected applications, as well as a yardstick by which the overall effort can be estimated.

The project approach is typically defined as follows:

1. Assess the volume of raw materials to be mined and determine reasonable units of work
2. Create an iteration plan by which those units of work will be addressed. The order should be dependent upon a balance of business value and functional dependency
3. Define measurable milestones by which each iteration will be assessed
4. Create a work-breakdown structure mapping specific tasks to the achievement of milestones
5. Assign resources to tasks and estimate time to complete each task

The resources needed for rule mining are not substantially different from those associated with the Business Process Modeling, Requirements, Analysis, and Design disciplines for Rule Harvesting. The team composition will typically include the following, at a minimum:

- Senior/Lead Decision Analyst(s), experienced in business rule mining, to lead the effort
- Support Decision Analyst(s), to work as scribes and to generate the large volumes of documentation prescribed by the leads.



- Business SME, to describe the business functionality and to validate the accuracy of the harvested rules
- Technical programmers, to assist in the analysis of source code and to write utility programs and/or automate repetitive code scans

5.3.4 Define Rule Mining Workflow

Enterprise rule mining is usually a multi-step process involving practitioners with disparate skill sets – including consultants, developers, architects, analysts, and subject matter experts. Often key personnel will be distracted by other projects and it is therefore crucial that a common workflow be defined and documented.

Following the guidelines provided further in this document, a high-level workflow may appear as:

No.	Step	Deliverable	Owner	Participant
1	Mine Candidate Rules	Candidate Rules	Team Lead	Developer Architect Systems Analyst
2	Verify Candidate Rules	Verified Candidate Rules	Systems Analyst	Systems Analyst
3	Transform to Business Rules Format	Business Rules Model	Business Rule Modeler	Systems Analyst Subject Matter Expert
4	SME Review	Approved Business Rules	Subject Matter Expert	
5	Report	Business Rule Reports	Subject Matter Expert	
6	Integrate	Rules within target toolsets	Architect	

Figure 36—Sample Rule Mining Workflow

Each step should be defined in detail, following an adopted methodology, project scope and constraints and rule mining technology usage. There may be multiple iterations of the first few steps until the rules are in an approved, final format.

5.3.5 Determine Business Processes Mapping and Alignment

The logic mined is important because of its business context. After all, logic is a subset of an overarching business process. Mined rules will make sense only when placed into context within the associated process.

Further, viewing applications in a business process context is important in order to identify priorities for rules mining activities. An enterprise commonly views itself in terms of its processes. Some of these processes are of critical importance to the organization, while others are simply commodity functionality.



Some processes may be meeting service level agreements set by line of business executives, and others not. Where modernization activities will focus will vary based on this calculus.

At this point, the business rule mining process begins to align with the decision analysis and business rule harvesting processes discussed in Section 4. The processes are indeed very similar. The main difference being: in decision analysis the SME defines the business rules and in rule mining the legacy code defines the business rules. Regardless, the decision analyst's job remains the same: organize, document and prepare the decisioning requirements for implementation are business rules.

Recalling the process outlined in Section 4:

1. Sketch out the high-level business process(es) using flowcharts or use cases
2. Identify the decision points within the business process(es)
3. Perform decision requirements analysis on each decision point, but in this case using the legacy source code as the source of business knowledge
4. Sketch out the DRD for the decisioning
5. Define the scope within the DRD

Once the business processes have been modeled as described above, application elements that support the chosen processes must also be identified. Here in lies the greatest risk to scope creep. In situations where the legacy application was organized in silos, this high-level match is a straightforward step. However, a monolithic order management application may provide a broad spectrum of functional services (i.e., customer enrollment, credit approval, order entry, and order fulfillment). If an organization is only interested in mining rules for one or two of those functional areas (i.e., the customer enrollment and order entry components), a mapping exercise between processes and their supporting application portfolio elements will be beneficial. Strict boundary discipline must be followed to prevent scope creep.

5.3.6 Perform Application Analysis

In the previous steps, an application has been inventoried and it has been understood, at a high level, where the required business processes of interest reside within application artifacts. The next task will be to decompose the application into its constituent logical elements.

This process is again very similar to that described for DRA in Section 4. The difference being that the primary sources for domain knowledge in DRA are business SME are replaced by the legacy codebase in rule mining.

A key factor continues to be contextualization. Elements of the application that are not relevant to organizational priorities are excluded. This scoping via context allows us to also see the 'boundaries' of rules and their associated impacts.

5.3.6.1 Application Decomposition

In this step, the application is further decomposed into its detailed components. The goal is to have a sufficiently detailed collection of artifacts to serve as input to the rule mining step.

Technology can help: Repository-based software can create a parse tree of detailed application objects and their relationships. This information is then presented in multiple graphical and textual views, synchronized with each other to facilitate context-sensitive analysis.

For example, a context view will display all data field declarations, procedures, and procedure calls within a program in a compressed and outlined mode. A detailed source code view will display code segments corresponding to the context view, allowing for quick navigation through the program via the context view. Traversal through the context view will enable a user to gain quick insights into a program's structure and complexity.

Other views available at the detailed program level include diagrammatic control flow between paragraphs, logic flowcharts within paragraphs, execution paths and runtime simulators for chosen



conditional outcomes. Such tools are used to gain detailed insights into an application prior to actually starting the rule mining phase.

Without the benefit of automated parsing tools, value can still be gained by conducting a manual inspection and walkthrough of application artifacts.

5.3.6.2 Identification of Exclusions

In the process of reviewing and analyzing an application, elements not to be included in the scope of rule mining, for functional and technical reasons, are marked. These may include standard utilities, reports, system routines and out-of-scope business processes. In the example, this would include the artifacts related to Credit Approval and Order Fulfillment, out of scope due to their nature as commodity, standardized business processes.

The identification of exclusions illustrates a benefit to be derived from the up-front business contextualization steps described above. If they had not been conducted, a "broad sweep" approach would have resulted in a higher investment at the rule mining and SME review stages, where rules from the commodity business processes would first have been mined and then later discarded as irrelevant.

5.3.7 Define Business Term Catalog

A major challenge with mining rules from applications is that it can be difficult to navigate the various variables and naming conventions within. These conventions have often a tenuous link to business terminology, and can make understanding the logic from a business perspective difficult.

A best practice is to refine these technical terms to create more a more business-centric view. This can be achieved through a business term catalog, which is a glossary of application objects and related business terms. Objects could be data fields, paragraphs, programs, data sources, and other application objects of interest.

Sources of information for a glossary of terms can be business documentation, data dictionaries, database schemas, user notifications, and even source code comments.

Automated rule mining tools offer a facility to propagate values for repeating patterns (commonly called 'tokens') within your application. For instance, the token 'ACCT-' may be replaced everywhere by 'Account-'. A tool would then use the glossary business names to replace technical terminology in the automated construction of candidate rules.

5.3.8 Define Rule Taxonomy

A Rule Taxonomy is a Rules Composition and Hierarchy Definition. The desired rules format is established in advance. The same rule to set an order discount may take alternate forms, such as

- *Declarative form:* "Each applicant who is a senior AAA member from California receives a 5% discount."
- *If-Then-Else form:* If an applicant is a senior, then if she is an AAA member, then if she resides in California, assign a 5% discount.
- *As an entry in a decision table:*



Condition	Result1	Result2	Result3
Senior?	Y	N	N
AAA member?	Y	N	N
Resident of CA?	Y	N	N
Action			
Order discount level	0.05	N	N

Figure 37—Decision Table Taxonomy

It can be useful to attach to a mined rule additional informational and workflow attributes, such as:

- Reviewer text annotations
- Rule type (I/O, calculation, data validation, security)
- Audit status (approved, not approved)
- Workflow status (extracted, working, accepted, rejected)
- Transition (valid, requires modification, duplicate, complete)
- Reviewer Identity
- Program derived from
- Code segment location (start, end)
- Code segment text
- Input and output data elements

A rule will also be placed within a hierarchy. All rules representing a decision or executing under a given set of conditions may be grouped into a Rule Set. Rule Sets will be grouped into higher level activity nodes reflecting the business processes they currently participate in.

If you are planning to populate an executable Business Rules Management System (BRMS), you will want to define a schema that is easily transferable into the specific target environment chosen. If the target environment is not yet known, refer to available business rule standards.

5.3.9 Mine Business Rule Requirements (a.k.a., Candidate Rules)

At this point rules are mined from the application artifacts mapped to the scope of the business processes identified in previous steps. Rule mining tools help you assure that excluded artifacts are not included in scope by enabling the organization of an application into sub-groupings. Rules will be mined for a sub-grouping and not for the entire application.

The specific rule mining approach taken is primarily driven by application patterns and the desired output.

5.3.9.1 Top-Down Approach

A top-down, or process-oriented approach starts from an examination of the user interface in an online application, or from the job flow in a batch application.

In an online application, a transaction may be invoked by a user selecting a menu option or entering a value to the screen. The fields that define the message or event that is sent from the screen to the interfacing application are identified. Each field in the triggering message may be considered a seed field for rule mining.

Using a seed field as a starting point, all of the downstream data impacts to the field including all conditional permutations are documented. Each data transformation (move into another field or calculation), represents a candidate business rule to be captured.

Rule mining software tools assists in this task by visualizing a data impact path forward for each seed field to each point where it is either populated by new values or used as input to other fields via comparisons, value propagation and calculations. At each such point, the tool can be used to document the underlying business rules. Automated rule detection methods can also be applied to capture each screen field edit as a candidate rule.

In a batch application, the concept is similar. Part of a job flow, e.g. a Job Control Language or group thereof that realizes a business process is identified, and all rules within individual programs relevant to that process are mined.

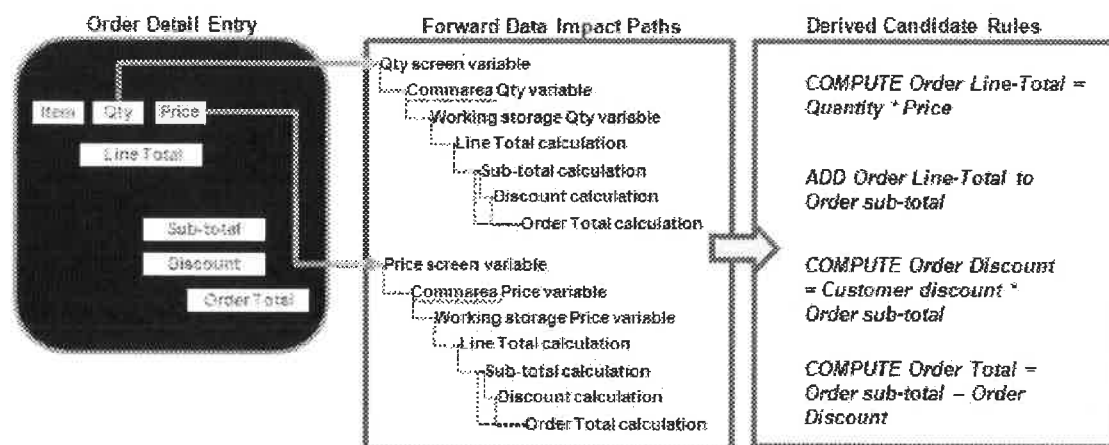


Figure 38—Top-Down Rule Mining

In the figure above, note the format of the resulting "Derived Candidate Rules". Automatically detected from a COBOL program, they resemble its constructs, with variable names replaced by the Glossary definitions. These will later undergo review and transformation to a more businesslike form. While this example is demonstrated for a COBOL application, advanced mining tools may apply to a broad array of languages from PL/ and Natural to Visual Basic and Java.

5.3.9.2 Bottom-Up Approach

A bottom-up, or data-oriented approach starts from an examination of system outputs – data sent to files (both batch and online), screens and output messages (online only).

Following this approach, rules are captured by starting from an interesting data point and identifying all logic impacting that point. For example, an Order Discount field is impacted by discounts calculated upstream from it, depending on the customer's location of residence.

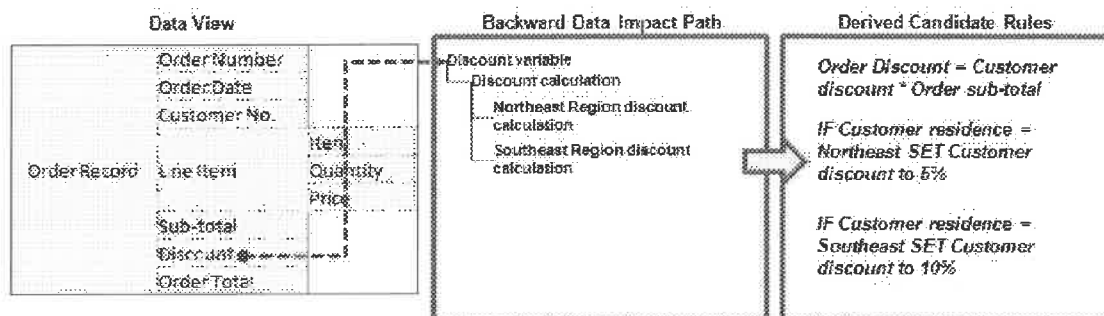


Figure 39—Bottom-Up Rule Mining

Rule mining technologies are particularly well suited to this approach. Through visual inspection or a repository query, data outputs of interest can be quickly identified. Then, automated rule detection routines are able to capture a candidate rule for each statement that impacts the point of interest. Because of the pre-organization into contextualized sub-groupings mentioned above, the search results will be constrained by the subset of business processes deemed relevant for rule mining.

Inspection of relevant DBMS tables may also produce rules embedded in keys and any data rules for referential integrity and value constraints. Once all data points of interest have been covered, all application logic of interest oriented toward existing outputs has essentially been mined.

5.3.9.3 Hybrid Approach

A hybrid approach combines the two approaches described above:

- The first step is a top-down oriented capture of the relevant transactions;
- For each transaction, bottom-up rule mining is performed, including only data outputs that have not yet been already mined for another transaction.

The benefit of this approach is to extend the coverage of rule mining while avoiding repetition.

Relating to the examples shown Figure 38 and Figure 39, following a strictly top-down approach resulted in repetitive efforts for the Quantity and Price fields since they both traversed identical downstream data impacts. Coverage was also partial since not all of the rules for Customer Discount were discovered.

Let's consider an extended case involving both Order Entry and Proposal Issuance processes. Adopting an exclusive bottom-up approach would have also resulted in repetition, mining rules for upstream data impacts that "hit" multiple outputs (e.g. customer discount rules). Using the hybrid approach, the first step is to mine rules from all outputs of the order transaction, then only outputs of the proposal transaction particular to it.

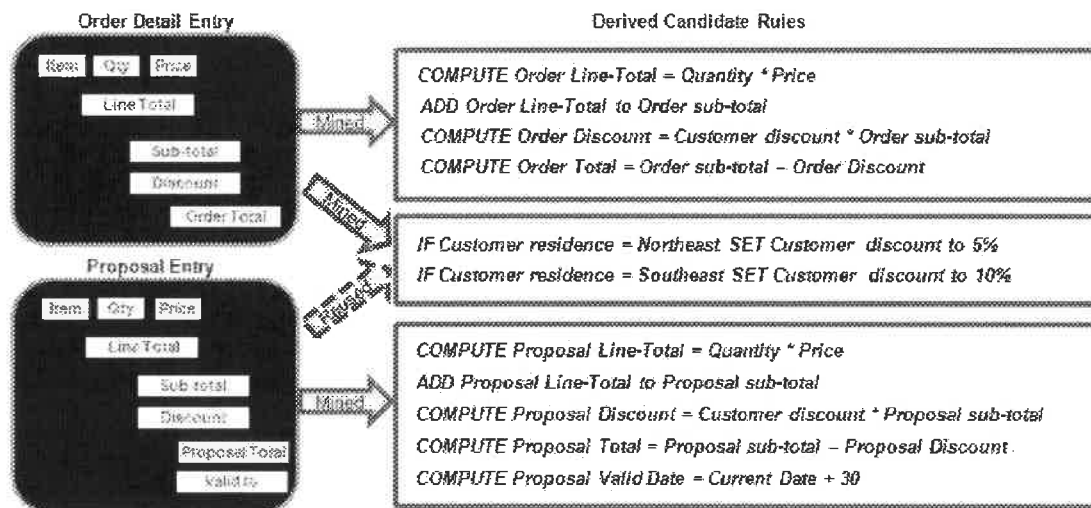


Figure 40—Hybrid Rule Mining

5.3.10 Verify and Validate Candidate Rules

At this point, after mining candidate rules from your application, verification and correction is a necessary step to ensure the correctness and completeness of the rules.

The candidate rules are examined for:

- Accuracy**—does each rule correctly reflect the underlying application behavior? If automated rule detection technology has been used, a rule at the point of interest (seed field) will be preceded by rules upstream from it, possibly with triggers, control conditions, and automatic rule set groupings. Each one of them (or a chosen subset) is reviewed for accuracy and corrections are made where needed, until the results are deemed satisfactory.
- Redundancy**—does a rule or rule set appear twice for the same application process? This can occur when rules are mined separately for two separate outputs that share upstream functionality. Or it can be a result of simple oversight like multiple team members inadvertently mining rules from the same code base. A rule attribute is used to mark duplication.

Another form of redundancy occurs when semantically identical rules were mined separately and with different names from different processes. This will be dealt with in the next step, when you transform candidate rules to business rules format.
- Completeness**—beyond predefined exceptions, has all of the application functionality been covered? A rule coverage report, matching mined rule sources to overall sources, can provide the answer.
- Relevance**—can each mined rule be considered a candidate business rule? Although this is not yet the SME review step, there may be certain constructs that, upon inspection, are clearly irrelevant and should not be included in the scope of rules for review. Security verification rules, housekeeping routines and out-of-scope operations may all fall into this category. Indicate relevance on one of the rule attributes.

5.3.11 Transform Business Rule Requirements

In the previous steps, candidate rules have been mined and reviewed, reflecting legacy application behavior. These rules closely follow the application's procedural flow and operations.

A transformational step is now required, to convert candidate rules to actual business rule requirements ready for review. This step is conducted either by application experts, rule architects, or subject matter experts. After review and conversion, the business rules captured reflect the current, as-is state to serve as a baseline or comparison to the target environment.

5.3.11.1 Reformatting to Business Rule Notation

If the candidate rules were constructed manually, they may already be in the chosen business rules format. In other cases, they may have been captured in a technical format (like cut and paste from source code) and will require some modification and regrouping.

If an automated rule detection tool was used, the resulting candidate rules may somewhat resemble business rules, by using the glossary definitions to place business names within rule names, data elements and controlling conditions. However, even after the rule verification step, most of the approved candidate rules will need to be adapted to conform to a chosen business rule notation.

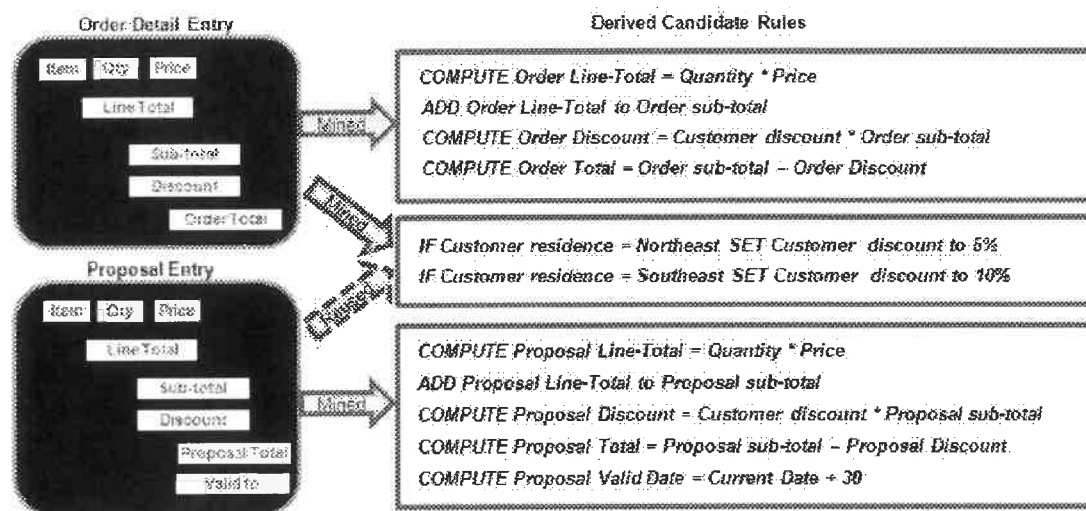


Figure 41—Business Rule Transformation

5.3.11.2 Fact Modeling and Rule Normalization

Due to their procedural nature, legacy applications tend to lock business logic into process-specific silos. However, true business rules are independent of process and should be maintained as such.

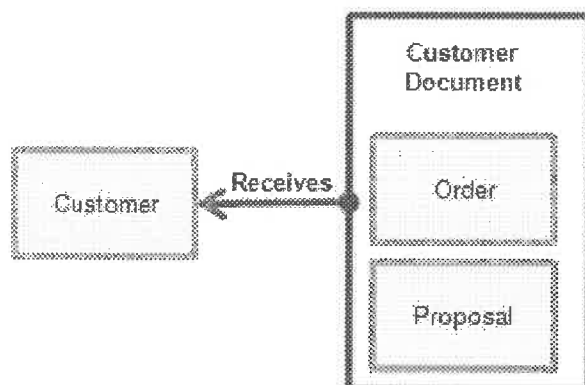
In the example, rules for the Order Detail Entry and Proposal Entry events have been separately mined and placed in Rule sets. Are they all unique? Upon further examination, most of the logic in them is identical by design. Analyzing the results from a business perspective, there is commonality between portions of any customer document – whether Order or Proposal.

From a tooling perspective, at this point it may make sense to switch over to a Business Rule Management System, importing the mined rules from the rule mining tool as described in the Integration section below.



Using a BRMS or a visual modeling tool, a Fact Model reflecting the significant business entities and their interrelationships discovered in your existing applications is constructed. These will link to the mined business rules and serve as a baseline for the to-be rules model.

In the example, part of the Fact Model would be:



Once this is done, the business rules are normalized to represent the desired business level semantics:

Normalized Business Rules	
Customer Document Handling	$\text{Discount basis} = \text{sum of (Item Quantity} \times \text{Item Price)}$ $\text{Discount} = \text{Customer discount percentage} \times \text{Discount Basis}$ $\text{Total} = \text{Discount Basis} - \text{Discount}$
Proposal Details	$\text{Proposal Valid Date} = \text{Current Date} + 30$
Customer Discounts	When Customer residence = Northeast, Customer discount percentage is set to 5% When Customer residence = Southeast, Customer discount percentage is set to 10%

...whereby the Customer Document Handling rules apply to both Orders and Proposals.

5.3.11.3 Grouping and Sequencing

At this point, the generated rule grouping and sequencing are considered. One point of attention is the triggering relationships between rules and other rules and rule sets. Since candidate rules are often derived from a 3rd generation language application (like COBOL or PL/I), they are automatically sequenced in a procedural manner. Transformation to a declarative mode will eliminate procedural elements that are non-business in nature.

As shown in the figure below, declarative relationships that reflect true business requirements will be modeled as triggers between rules and other rules or rule sets. In the majority of BRMS environments, a single rule may trigger multiple rules and rule sets, where the sequencing of each triggered rule or rule set is pre-compiled or resolved only at runtime.

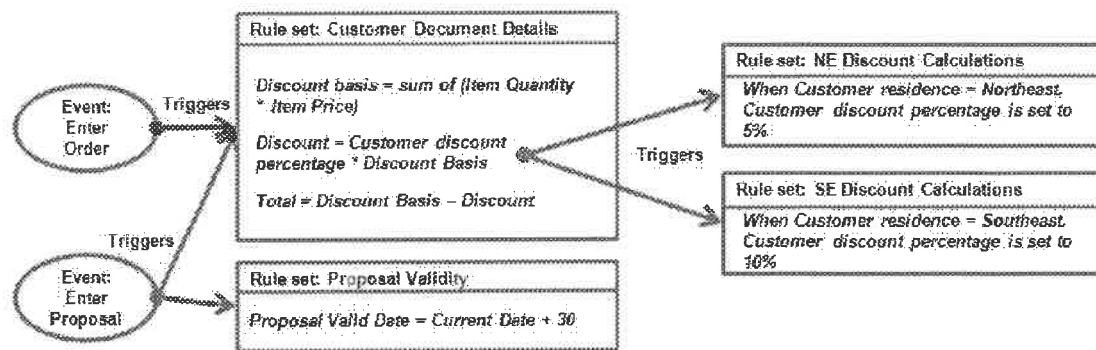


Figure 42—As-is Business Rule Model (Event-driven)

5.3.11.4 Subject Matter Expert Review and Approval

Once mined rules have been transformed into business rules, they are handed over to subject matter experts (SMEs) and / or business analysts for review and approval.

Normally, SMEs will not make major changes to the rules at this point. Rule mining tools may include rule attribution capabilities to aid the SMEs and enable them to mark up the business rules as

- Approved or rejected;
- Reclassified to another rule category within the taxonomy;
- Annotated with additional information in textual description attributes.

Rule mining tools also often offer web portals with a functional focus on predefined SME activities. This can greatly accelerate the review and approval process.

5.3.12 Reports Production

Business rule reports are created in either hardcopy or digital formats. Rule mining tools produce reports and diagrams depicting detailed or summary rule information within your chosen context: hierarchy level, grouping, search result. These reports serve both as reference in the review steps and as documentation of record.

5.3.13 Integration with a Target Environment

Depending upon the adopted modernization strategy, integration requirements with other environments will vary.



5.3.13.1 Business Rule Development

Redevelopment with a business rule approach will typically leverage a BRMS authoring environment. These tools typically include XML import capabilities, which will be used to define an "as-is" business rule space, allowing rule developers to selectively re-use candidate rules deemed relevant for the target environment. This will provide valuable (and sometimes crucial) traceability from newly deployed rules back to their legacy origins.

5.3.13.2 Conventional Development

This approach typically involves building Java and .NET applications with comprehensive developer toolkits. Often, UML models will be used to define logical application views prior to actual code generation.

In these environments, mined business rules can be attached as behaviors in UML classes that leverage them. For example, an Order_Invoice class which includes Order_Discount as an attribute may also include Calculate_Order_Discount as a class behavior. This behavior can be derived (and potentially imported) from the mined business rule performing the same function.

5.3.13.3 BPM

Process Management Business Process Management (BPM) tools facilitate process model creation and linkage to underlying rules and executable services. They also include the ability to define workflow rules (using BPEL) to govern the manual and automated transitions between activity nodes.

In this context, each activity node may be realized by business rules. Many-to-many relationships may exist between rule sets and supported activities. Populating BPM processes with their relevant mined "as-is" business rules can "close the loop" for business analysts and significantly advance IT / business alignment goals.

5.3.13.4 Requirements Management

Vendor offerings also include requirement tools that enable the definition of high level use cases, detailed flowcharts and activities for effective application development and management. In these environments, mined business rules can be imported and attached as either core requirements or as textual annotations to activity nodes.

5.3.13.5 SOA Enablement

Service enablement of existing applications involves code refactoring and deployment as service capsules. The rule mining step can be invaluable in locating fine-grained services within source code and serving up the required service components. For example, the results of automatic rule detection for the calculation of an order discount will include all code segments leading up to the final calculation. By creating a component slice with that code (and its dependents) only, the order discount calculation can be redeployed as a service.

SOA Enablement is important regardless of whether the actual decision services are fine or course grain services. Regardless of whether the individual fine-grain rulesets are meaningful as standalone application components, the fine-grain rulesets may be reused as components within other decision services.

5.3.14 Proactively Manage Rules

Most applications are expected to continue and maintained for many years into the future. Having mined rules from them, it is crucial that they continue to be updated and kept synchronized with future application changes. Rule mining tools offer maintenance and management capabilities, including

- Automatic alignment of rules with their original code segments even when they have moved as a result of overall source code changes;



- Audit trails for manual rule changes;
- "Changer" routines allowing for individual or mass changes, post- rule mining.

5.4 Rule Mining Recommendations for Chubb

The following are recommendations to help Chubb be successful in their rule mining efforts.

5.4.1 Some Rules of Thumb about Rule Mining

Mining decision requirements form an existing codebase is not the recommended approach. Mining should only be undertaken if other sources of knowledge are otherwise unavailable or extremely limited.

Rule mining is not a substitute for, and does not forgive the need for, doing proper decision requirements analysis.

Before undertaking a rule mining effort, there are some guidelines and recommendations to consider.

1. Focus on the fundamental goals of a BRMS: flexibility and ease of maintenance and extension of the business rules.
2. Do not pave the cow path.
3. Think "Diamond Mine" not "Coal Mine."

5.4.1.1 Think "Diamond Mine" Not "Coal Mine"

Forgetting the third guideline is the most common mistake made in mining rules from source code. The rule mining metaphor creates connotations of a large coal mining operation where large quantities of usable materials are extruded from the ground and pretty much used as is and on mass. That is not the case for rule mining. Rule mining is more akin to mining gems. Gem must be painstakingly exposed from within comparatively large volumes of debris. And, gems are typically not directly usable in the state they are mined from the ground. After being mined, gems must be cut, polishes, and then mounted.

Rule mining is a substitute for the SME, not the decision requirements analysis process. It is best to think of rule mining as "rule requirements mining." The goal is not to discover the business rules buried within an existing codebase. The goal is to identify the business decisioning requirements that are implemented within the existing code base. These requirements then feed a traditional decision requirements analysis process very similar to that described in the previous section.

5.4.1.2 Do Not Pave the Cow Path

The second most common mistake made in mining rules is to lose focus on process-improvement. It is very easy to fall into a mode where you re-platform the existing codebase within the BRMS. While doing so may have advantages in expediting the implementation phase of the project, doing so is rarely advantageous to the resultant BRMS.

When doing rule mining, rule analysts must be mindful that the previous implementation was designed to follow the conventions and practices of a different generation, a different technology, and a different set of project goals and success criteria. Some examples include:

1. COBOL code is typically structure to follow a flow chart or transactional use case. Business rules that apply to multiple transactional flows are very often replicated in multiple modules. And, they are usually coded as they are applicable to specific cases, which can make a single business rule appear to be multiple rules.



2. COBOL modules are not services-oriented and are almost always tightly coupled with the business data. As a consequence, business rules are often intertwined with query logic and data validation/verification checks.
3. Mainframes are very efficient at managing data. It is common for seasoned COBOL developers to resolve decisioning logic down to data lookups. This often disguises the true business decisioning and can make a relatively small number of rules appear quite voluminous.

In a way, this guideline precipitates from the prior guideline about rule mining being a substitute for the absence of a SME, not for the proper analysis of requirements. So, allow time to perform proper analysis on the mined decision requirements. Do not simply specify the new implementation of rules as per the existing codebase.

5.4.1.3 Focus on the Fundamental Goals of a BRMS

What are the goals of a BRMS project?

- Deliver fast, accurate, and consistent decisions across functions, channels and customer touch points
- Organize business rules in a central repository for reuse & easier management
- Let business users quickly & safely update rules to react fast to changing conditions
- Verify, validate & simulate changes to ensure rules meet business needs
- Test new strategies & learn faster using champion/challenger & adaptive control
- Leverage predictive models & optimization to continuously improve decisions to better achieve business goals

What are the benefits to the business of achieving the above goals?

- *Precision*—Business decisions are precise and targeted.
- *Consistency*—Decisions are consistent across all channels, business units, and geographies.
- *Agility*—Business can adapt quickly to dynamically changing conditions
- *Speed*—Decisions are executed in real time
- *Cost*—Efficiencies in maintaining and extending decisions

The third most common mistake that precipitates from rule mining is a loss of focus on the reasons why a BRMS technology was selected for the project in the first place. No reasonable person starts a BRMS project with the expectation of being able to front-end load “rule ore” into a business rule engine and achieving the above goals. Yet, most rule mining projects find themselves in that situation about half way through the effort.

The success of any BRMS project depends upon doing the proper upfront analysis and design, and following a BRMS methodology throughout the SDLC.

5.4.2 Manual Rule-Mining Recommendations

The documents and workbooks that FICO uses for rule mining are precisely the same as those used for rule harvesting. FICO has provided Chubb with templates for these artifacts and they are available through Chubb's BRMS COE.

Recommendations include:

4. Allow the taxonomy for the workbooks to evolve as rules are mined.
5. Allow the DRD to evolve through iterative elaboration as rules are mined



6. Favor a hybrid strategy as opposed to strict top-down or bottom-up.

5.4.3 Subject-matter Experts (SME)

Endeavoring to build a BRMS without a SME is starting out from a disadvantaged position. Therefore, it is very important to ascertain that there is absolutely no access to subject-matter expertise.

FICO has helped hundreds of customers build BRMS applications and has found that it is indeed very rare for an enterprise to outright lack expertise in any aspect of its business. More commonly, it is because of one or more of the following perceptions than actual reality:

1. IT has unrealistic expectations about the level of knowledge required of a SME
2. The SMEs do exist, but they are just unknown or unavailable to IT
3. IT does not recognize that their analysts are qualified SME

FICO strongly recommends that Chubb find and involve SMEs as much as is possible on the rule mining efforts

5.4.3.1 All-knowing Sage vs. Expert

The most common misperception about the availability of subject-matter expertise comes from unrealistic expectations about the level of knowledge required of a SME. Often the SME is envisioned as an unquestioned authority on everything and about everything related to the business domain of the application. The perception is that if such an all-knowing sage does not exist, then a SME does not exist. The reality is that such expectations are not realistic. Moreover, practice has shown that the more of a guru the SME, the more theoretical they tend to be, and thus the less valuable they are to DRA process.

A SME simply needs to have a comprehensive understanding of a business domain. For example, consider a certified public accountant. A CPA would certainly be considered a SME with respect to applying tax code to the filing of tax returns. They certainly do not have every rule and regulation memorized. They understand the principles of the tax code; they know how to apply those principles, and know how and where to look up the detailed minutia.

5.4.3.2 Availability vs. Lack of SME

As mentioned earlier, FICO has found it to be quite rare for business to lack subject-matter expertise. However, it is not uncommon for business subject-matter expertise to be unavailable to IT. This may be for cultural reasons, i.e., a client/contractor relationship between business and IT. This may be because the business is engrossed in other priorities and does not have the bandwidth to support IT. This may be because the source for business subject-matter expertise is unknown to IT.

In many cases, IT management can escalate requests through the proper channels to make business SME available to IT. If at all possible, FICO recommends this be done. However, if business SMEs are truly unavailable, then the IT does in fact lack support from SME and rule mining without SME support may be the only alternative.

5.4.3.3 Business Savvy IT/IS Analysts

This is sort of a corollary to the first misperception, but common-place enough to stand on its own. FICO has seen that the IT/IS analysts within many corporate IT departments are actually quite knowledgeable about the business subject-matter behind their business software systems. These analysts are qualified as SME and the IT department is capable of performing DRA with little involvement from the true business SME.

It is important to delineate between IT/IS analysts who are SME and those who are just power users of existing systems. Being a power-user, meaning understanding how to use the existing system, is not



the same as being a SME. To be a SME, the IT/IS analysts must understand the business decisions that are implemented in the business systems.

5.4.4 Subject-matter Expertise Is Needed for Rule Mining

Bottom line: SMEs are still needed for rule mining. They may not be as intimately involved in the workshop process as DRA. They may not function as the source of authority for the business knowledge behind the decisioning requirements. Nonetheless, they are indeed critical to the success of the rule mining effort.

The above is true regardless of the implementation technology being applied. Requirements are requirements, regardless of whether they are implemented using a BRMS technology like Blaze Advisor or simply coded in Java. The mining effort is still needed and the SMEs are an important part of the mining effort.

5.4.4.1 Questions and Clarifications

In the simplest sense, SME are needed to assist the decision analysts and technical programmers with questions and clarifications about the functional meaning and relevancy of the mined decision requirements. Otherwise there is a risk that the rules may be technical rules rather than business rules. Further, there is a risk that the rules may not be of significant value to the business.

5.4.4.2 Attributes of a Quality Requirement

This checklist provides guidance on assessing the quality of requirements. The first 4 attributes (correct, complete, clear, and consistent) are best assessed by a SME.

- ☒ *Correct* – Does the requirement correctly specify a true need, desire, or obligation?
- ☒ *Complete* – Is the requirement stated as a complete sentence?
- ☒ *Clear* – Is the requirement unambiguous and not confusing?
- ☒ *Consistent* – Is the requirement in conflict with other requirements?
- ☒ *Verifiable* – Can we determine whether the system satisfies the requirement?
- ☒ *Traceable* – Is the requirement uniquely identified?
- ☒ *Feasible* – Can the requirement be satisfied within cost and on schedule?
- ☒ *Design Independent* – Are all requirements that impose constraints on the design, limiting design options, justified?

5.4.4.3 Verification and Validation of Candidate Rules

After mining candidate rules from the application, the mining process requires verification and correction is a necessary step to ensure the correctness and completeness of the rules.

The candidate rules are examined for:

- *Accuracy*—does each rule correctly reflect the underlying application behavior? If automated rule detection technology has been used, a rule at the point of interest (seed field) will be preceded by rules upstream from it, possibly with triggers, control conditions, and automatic rule set groupings. Each one of them (or a chosen subset) is reviewed for accuracy and corrections are made where needed, until the results are deemed satisfactory.
- *Redundancy*—does a rule or rule set appear twice for the same application process? This can occur when rules are mined separately for two separate outputs that share upstream functionality. Or it can be a result of simple oversight like multiple team members inadvertently mining rules from the same code base. A rule attribute is used to mark duplication.



Another form of redundancy occurs when semantically identical rules were mined separately and with different names from different processes. This will be dealt with in the next step, when you transform candidate rules to business rules format.

- **Completeness**—beyond predefined exceptions, has all of the application functionality been covered? A rule coverage report, matching mined rule sources to overall sources, can provide the answer.
- **Relevance**—can each mined rule be considered a candidate business rule? Although this is not yet the SME review step, there may be certain constructs that, upon inspection, are clearly irrelevant and should not be included in the scope of rules for review. Security verification rules, housekeeping routines and out-of-scope operations may all fall into this category. Indicate relevance on one of the rule attributes.

The above is best performed with the support of SME.



6 Rule Taxonomy Recommendations for Chubb Personal Lines (CPI)

Taxonomy is, by definition, the practice and science of classification. The purpose of taxonomy for business rule is simply to document the fact that This_Rule is a This_Kind of rule and That_Rule is a That_Kind of rule during the rule discovery and analysis process.

The term "Rule Taxonomy" is very loosely applied in the BRMS industry. Any sort of schema for classifying or organizing business rules as they are discovered will usually end up being called a Business Rule Taxonomy by the methodologist who devised the schema. However, taxonomy is not supposed to be used as an approach to organizing rules, just classifying them.

There are dozens of different views on rule taxonomy and the subject is debated with much fervor amongst the BRMS methodologist. The purpose of this whitepaper is not to take sides or recommend one over another. The goal of this paper is simply to provide an overview of the approach and to provide Chubb with guidance in selecting one.

6.1 Overview

The subject-matter covered by the subsections under this section include:

4. There are Dozens of Rule Taxonomies, and They Aren't All Wrong
5. FICO's Approach to Rule Taxonomy
6. FICO Recommendations for CPI Rule Taxonomy
 - a. FICO Recommendations for Underwriting Taxonomy
 - b. FICO Recommendations for Print Taxonomy

6.2 There are Dozens of Rule Taxonomies, and They Aren't All Wrong

A little quick research will show that there are literally dozens of published rule taxonomies available within the BRMS industry. They are available both free (open) or for a fee. But they all seem to have one thing in common: they are all better than the rest because they either fix something that is wrong with a widely-accepted methodology, or they are the newest and keenest mousetrap.

6.2.1 Enterprise/Organizational Taxonomy vs. BRMS Application/Project Taxonomy

Table 8 shows some of the more popular methodologist and a summary of their recommendations for structuring taxonomy. The table seems quite extensive and, considering that it lists only the most popular or commonly followed taxonomies, is somewhat overwhelming. But in reality, they are all variation on a few themes



Table 3—Rule Taxonomy Recommendations by Methodologist

Methodologist	Taxonomy Recommendation
Guide Business Rules Project	<p>Classification is broader than just rules and applies to all harvested knowledge about a domain.</p> <p>Taxonomy is based upon:</p> <ul style="list-style-type: none"> • Structural Assertion (terms, facts) • Action Assertions (integrity constraints, conditions, authorizations) • Derivations (calculations, inferences)
Ron Ross Database Research Group	<p>Classification is broader than just rules and applies to all harvested knowledge about a domain.</p> <p>Taxonomy based upon:</p> <ul style="list-style-type: none"> • Facts • Terms • Rules (constraints, derivations, inferences, timing, sequence, heuristics)
James Odell Independent Consultant	<p>Intended for the classification of rules only. Classifies rules by the role each plays in the decisioning proces.</p> <p>Taxonomy based upon:</p> <ul style="list-style-type: none"> • Constraints: (stimulus/response, operation, structure) • Derivations: (inferences, compilations) <p>Note: rules are further classified as global, local and/or temporal</p>
Tom Romeo IBM	<p>Intended for the classification of rules and data. Classifies rules by the role each plays in the decisioning process.</p> <p>Taxonomy based upon:</p> <ul style="list-style-type: none"> • Structural (relationships, domains), cardinality, optionality, • Behavioral (pre-conditions, post conditions, derivations)



FICO Recommendations Whitepaper for Chubb Business Rule COE

Methodologist	Taxonomy Recommendation
Margaret Thorpe, Tangram	<p>Intended for the classification of rules w/ definitions. Classifies rules by the role each plays in the decisioning process.</p> <p>Taxonomy based upon:</p> <ul style="list-style-type: none"> • definitions • basic integrity constraints • general declarative constraints • procedural constraints • inferential • derivation
Barbara von Halle, Knowledge Partners	<p>Classification is broader than just rules and applies to all harvested knowledge about a domain.</p> <p>Taxonomy based upon:</p> <ul style="list-style-type: none"> • Definitions • Facts • Constraints • Derivations • Inferences
Usoft Corp.	<p>Intended for the classification of rules only. Classifies rules by inherent attributes of the rules.</p> <p>Taxonomy based upon:</p> <ul style="list-style-type: none"> • Restriction (constraints, domain) • Deduction • Behavioural • Representation
Dan Tasker, Air New Zealand	<p>Intended for the classification of rules only. Classifies rules by inherent attributes of the rules.</p> <p>Taxonomy based upon:</p> <ul style="list-style-type: none"> • Action Restricting • Action Triggering • Constraint

March 31, 2011

FICO Confidential Page 152 of 259

Confidential

FED007129_0152



Methodologist	Taxonomy Recommendation
Brightware	<p>Intended for the classification of rules only. Classifies rules by the role each plays in the decisioning process.</p> <p>Taxonomy based upon:</p> <ul style="list-style-type: none"> • Business Rule • Policy Rule • Workflow Rule • Decision Heuristics (meta-rule)
Vision Software	<p>Intended for the classification of rules only. Classifies rules by the role each plays in the decisioning process.</p> <p>Taxonomy based upon:</p> <ul style="list-style-type: none"> • Validation Rules • Derivation Rules • Relationship Rules • Conditional Actions

Part of the reason for there being so many taxonomies in the BRMS industry is that the taxonomy is used to credentialize the methodologist. This whitepaper is not going to get into which methodology is better than others, or in any recommend one approach over another. Bottom line: any organization will be successful in BRMS using any of the above approaches, as well as any of a dozen others not listed above.

6.2.2 Classification vs. Organizational/Structural

Within the blogs from the BRMS industry thought-leaders, there are discussions and recommendation regarding the use of Taxonomy for defining the organization of rules and/or the structure of rules within the repository. In the opinion of this author, taxonomy should be pure classification. Organization is more like Ontology. Here are the differences.

Taxonomy is like pigeonholing. Any item can only belong to one taxon. i.e., an animal cannot be both vertebrate and invertebrate, or mammal and fish, or dog and wolf. Classification is based on an analysis of certain important attributes of the things being classified. It typically has nothing to do with organization or structure and if used as such may actually complicate finding objects within the structure. Taxonomy relationship between the things being classified into taxon, as well as hierarchies taxons, is called an "is-a" relationship because it easily identified by use of "is a" or "are" in English: e.g.s. Tobi **is a** dog. Dogs **are** mammals. Mammals **are** vertebrates. Now by understanding the characteristics that define membership in each of these taxon, we inherently know a lot of information about Tobi.

Ontology is a rigorous and exhaustive organization of some knowledge domain that is usually hierarchical and contains all the relevant entities and their relations. Moreover, that organization usually has nothing to do with the classification. e.g. When identifying animals that would make good pets, the selection of a dog vs. a wolf, a Guinea pig vs. a shark, or a goldfish vs. an earthworm has little if anything to do with the taxon to which each type of animal belongs.



Furthermore, in ontology, something can belong to one, more than one, or zero groups or collections. i.e., Danny is a 13-year-old male human (Danny's Taxonomy). Danny is in 8th grade, is a Boy Scout, plays in the Middle School Band, and belongs to the Religious School Youth Group (Danny's Ontology).

Just like Taxonomy, we inherently know a lot of information about Danny just by understanding the groups and organizations that he belongs to. It is easy to see that Danny's ontology would be an effective way to structure and organize knowledge about Danny and his behavior.

There are times, however, when similar attributes are used in both classification and organization. This is where the confusion and overloading comes into play. Indeed, it is OK to classify and to segregate on the same attribute. This is done all the time. In the example above, we see that Danny's membership in the group of 8th-graders is tightly coupled to his age because (for the sake of this example, assume) every member of the 8th-grade is a 13-year-old human. We see his participation in the Boy Scouts as directly coupled with his gender being male.

The problem arises when the ontology is extended. Danny's membership in the Middle School Band is loosely coupled age because Band is open to members of the 5th, 6th, 7th, and 8th graders. So, it is really his membership in a grade, which is implied by his age... But, not all 8th-graders are members of the band... You can see how this would get complicated if his classification were used to organized information about being in the band.

The goal of ontology is to organize rules in a way such that they are easy to maintain and extend. But, they also need to be easy to find in the repository and intuitive for packaging for deployment. Ontology is not a knee-jerk decision to be made. It needs to be carefully considered and flexible to changes as more is learned about the domain during discovery and analysis.

6.2.3 Taxonomy for the Enterprise or Application/Project

Opinions about the importance of a taxonomy varies depending upon who you talk to, their role in the BRMS competency model, and the scope of their vision within an organization. For instance:

1. Rule Taxonomy is indispensable to a Principal Decision Analyst working with an organization to harvest business rules at an enterprise level. They are usually tasked with mulling over tremendous volumes of organizational knowledge and they need a well-defined and somewhat rigid method for classifying that knowledge so that they can make sense of it.
2. Rule Taxonomy is critical to any effort to harvest organizational knowledge using BRMS methodologies intended to produce artifacts that are BRMS platform agnostic.
3. Rule Taxonomy a very important tool for a Senior Decision Analyst leading a team of junior Decision Analysts on a BRMS project, especially if the junior decision analysts are working on their first BRMS project. The taxonomy will help the junior analysts learn to differentiate between true business rules and organizational knowledge that is actually meta-data, derivational, relationships, etc.
4. Rule Taxonomy is useful to the Rule Architect for a BRMS project. He could certainly live without it because his SOA and OOA/OOM background will give him an intuitive understanding of how to classify the harvested organizational information. However, it is useful and will save many Q&A sessions if this is provided to him as part of the harvesting documents.
5. Rule Taxonomy is not very important to a Rule Developer who writes rules and conducts unit-tests. The actual writing of rules for an inference-base decision platform is not as concerned with taxonomy. By the time the rule requirements get down to the developer's level, any consideration for classification should have already be accommodated by the rule architect.



6.3 FICO's Approach to Rule Taxonomy

FICO is in the business of providing Decision Management Solutions and Technologies (DM Applications and DM Tools) to help our customers implement and manage decisioning systems within their enterprise. FICO Professional Services provides consulting services around the same.

FICO has encountered decision requirements that were prepared by just about all of the leading methodologists in the industry. Our architects have learned to deal with a number of approaches and see strengths and weaknesses in most methodologies. FICO's view of rule taxonomies is more or less aligned with the opinions expressed in Malcolm Chisholm's November 2004 blog on "Rethinking Rule Taxonomies" (<http://www.information-management.com/news/1014487-1.htm>). Chisholm more-or-less said that taxonomies are tools. Pick one that you like and that you feel will work well for your application, and move on.

6.3.1 Guidelines for Selecting a Rule Taxonomy

To assist Chubb in the selection of a Taxonomy, FICO offers the following recommendations:

1. Select a mature and proven Taxonomy. There are numerous taxonomies available from the numerous thought-leaders in the BRMS industry. Unfortunately, many are new and unproven and many were conceived by theorists who lack the experience of hands-on practice.
2. Select a taxonomy intended is for classification only. Do not go for one that combines the concepts of classification and structure/organization. It makes the classification schema more complicated and less broadly applicable.
3. Select a rule taxonomy in which the classification criteria are defined such that any give piece of knowledge will belong to one and only one taxon. If the taxonomy is hierarchical, each rule will belong to one and only one leaf-level taxon.
4. Manage the ontology of knowledge separate from its taxonomy.
5. Pick one approach and stick with it. Cohabiting multiple taxonomies based on multiple theories will complicate the sharing and reusability of BMRS artifacts across an organization.

6.3.2 FICO's Rule Taxonomy

FICO's rule taxonomy is intended for classifying the artifacts generated from rule harvesting at the application or project level. Details on this taxonomy are provided under Section 6.9, Decision Harvesting. This section provides a high-level overview.

FICO taxonomy classifies the outputs from Decision Harvesting according to:

- Business Terms
- Decision Dependencies
- Rules
- Abstractions
- Computations

FICO's decision harvesting workbook templates provide a separate worksheet for each. See Section 6.9 for details.



6.4 FICO Recommendations for CPI Rule Taxonomy

FICO does not endorse the rule taxonomy of any one methodologist over another. FICO has a track record of success in implementing BRMS solutions from rule harvesting artifacts that followed all of the approaches listed in Section 6.2.

The PVDS project has developed a rule taxonomy that is an adaptation of the FICO Taxonomy. They achieved a high degree of success with it and found it to be very effective for their purposes. FICO feels that it would be very reasonable to start with the PVDS Taxonomy and adapt it to cover the larger Chubb enterprise.

It is important to note that individual projects may not produce rule harvesting artifacts that fit into every taxon in the taxonomy. i.e., the CPI Print project may not have facts (which is a very common taxon in several methodologies).

6.4.1 FICO Recommendations for Underwriting Taxonomy

There are methodologies that tend to produce very rich requirements specifications when applied very early in the SDLC. Conversely, these methodologies are a little less well-suited for projects that are further along. The underwriting team is really just getting started. They would be well-served by following one of the more rigorous methodologies like:

1. Ron Ross
2. Barbara von Halle
3. Margaret Thorpe

Ron Ross's methodology is one of the most rigorous and classifies decision harvesting artifacts according to the following schema:

- Terms
- Facts
- Rules (and then into one of the following rule types)
 - Constraints
 - Derivations
 - Inferences
 - Timing
 - Sequence
 - Heuristics

Separating timing and sequence into two types is not necessarily beneficial for Blaze Advisor because the ruleflow metaphor is used for both. It is often advantageous to combine timing and sequence into Business Process.

6.4.2 FICO Recommendations for Print Taxonomy

CPI's print project is in an advantageous position when it comes to the selection of a rule taxonomy. The more that is known about a domain, the more specific the taxonomy can be. Most enterprise taxonomies are very broad in classification. Print has the opportunity to implement a very specific and rich classification schema.



6.4.2.1 Taxonomy for Print

FICO recommends the following classification schema as a starting point for classifying rule harvesting artifacts for the Print project:

- **Terms**—Elements of data or information that is required for the print process. If not all the terms are required, then the addition of Required and Optional subclassifications will be useful.
- **Facts**—Assertions of truth about the domain that are meaningful to the decisioning process. None are known at this time and it is likely that Print will not need this taxon.
- **Rules** (and then into one of the following rule types)
 - **Validations**—verification that the required data is present and valid with respect to itself and with respect to other data. i.e., the insurance product is specified and is valid with respect to the residence state.
 - **Document selection rules**
 - **Content selection rules**
 - **Content sequence rules** (if required, this may be delegated to the content management layer)
 - **Business process**—a rule about the timing or sequencing of one rule or set of rules relative to another rule or set of rules. There are several kinds of meta-rules. The Print team is probably too early in their learning curve to make these subtle determinations. Therefore, it might be best to omit the detailed classifications below and leave it at the BP level.
 - **Ruleflow**
 - **Meta-rule**—a rule about rules. This is a type of business process rule where decisioning is performed about a decision. None are known at this time and it is likely that Print will not need this taxon.
 - **Heuristics**—a rule of thumb that is not necessarily a firm true or false. Often in regard to meta-rules. i.e., if power is off to all the outlets in a circuit, and the circuit breaker is on, then check for a GFCI. This specifies a strategy to pursue in a diagnostic process that may or may not be correct and is dependent upon special circumstances. None are known at this time and it is likely that Print will not need this taxon.
- **Dependencies**—logical dependencies between decisions. i.e., some decisions depend upon the outcome of others. Some decisions are made differently depending upon which data elements are known.
- **Abstractions**—abstract concepts that can be used to replace series of repeated conditions. i.e., if the pattern of conditions
 - If State is ME or State is NH or State is VT or State is MA or State is RI or State is CT is repeated in multiple rules, it may be appropriate to specify one abstraction rule that says:
 - If State is ME or State is NH or State is VT or State is MA or State is RI or State is CT then Region is New England
 and use that simple predicate as a proxy for the multiple state comparison.
- **Computations**—simple calculations or algorithms that are not really decisions, yet are needed for the decisioning process



6.4.2.2 Ontology for Print

As far as organization goes, FICO's prior experience with Forms and Document Management projects tells us that the best organization for harvested knowledge is around the individual forms. The following is an ontology that worked well on another similar project for forms:

- Required Data Validations
- Form Family (collections of similar or strongly-coupled Forms)
 - Form (for each Form in the Form Family)
 - Form Selection
 - Content Selection
 - Endorsements/Riders/Clauses
 - ERC Selection
 - Content Selection
- Form (for each standalone Form)
 - Form Selection
 - Content Selection
 - Endorsements/Riders/Clauses
 - ERC Selection
 - Content Selection
- Results—Packaging and post-processing

Print can adapt the Forms concept to their Policy Document model and define a similar schema. It might be as simple as replacing the concept of a Form with the concept of a Document and rolling up the concepts of Endorsements/Riders/Clauses with Documents.

FICO believes that organizing rules by their applicable forms will produce the most maintainable and extensible solution. It is not important that rules about multiple forms use the same data elements (terms). The inference engine manages that in the business object model in working memory. That said, if organizing the rules within the Document groups by data elements simplifies the documentation, it could be a nice addition.

It is important to reiterate that the goal of ontology is to organize the rules so that they are easy to maintain and extend. FICO recommends Chubb engage FICO PS to review both their Taxonomy and their Ontology prior to investing a substantial effort into mining rules.



7 Personal Lines Insurance Rule Use Cases

This section provides several executive-level use cases for hypothetical scenarios for the application of BRMS for Chubb Personal Insurance's Underwriting. Most of the scenarios in this section come from strategies and real-world implementations by FICO customers in the Personal Lines Insurance industry. FICO Sales and Marketing can provide further elaboration on selected content.

7.1 Overview

The subsections of this of this chapter cover the following scenarios for the application of BRMS within Personal Lines Underwriting.

9. BRMS and Underwriting for Personal Lines Insurance
10. New Focus for IT/IS: Revenue Generation
11. Making Revenue-Generating Decisions at Point of Sale
12. Getting a Better Handle on the Channel
13. Multiplying Profits and Markets through More Accurate, Innovative Pricing
14. Facilitating Profitable Renewals
15. Expanding the Market by Tightening the Market
16. Decision Management Is the Next Step

7.2 BRMS and Underwriting for Personal Lines Insurance

*New Strategies for Cost control and Pricing Precision
using FICO Decision Management Technologies and Services*

Today Personal Lines Insurance carriers are extending the efficiencies achieved through back office automation to point-of-sale (POS) decision making across all channels, including self-service websites, in-house sales and agent/distributor networks. New Decision Management solutions not only facilitate straight-through-processing, by improving the quality of inbound data, they also increase the accuracy, objectivity and consistency of underwriting and application decisions, even when business volume is rising rapidly. In addition, by providing deeper insights into loss risk, Decision Management enables carriers to expand risk tiers, assign more accurate pricing and rapidly bring innovative products to new markets.

These improvements in POS decision making are helping PL carriers substantially increase the amount of new business coming in through channels, with a high degree of confidence in the value of that business. Because Decision Management provides an objective measure of potential loss and profit, it can also help in evaluating channel performance and improving channel management. Moreover, better application data means underwriting and new business staff no longer spend time correcting errors and chasing down missing information, which not only cuts costs but also frees up resources to focus on improving producer relations and results.

Decision Management, which works with existing rating engines, underwriting applications and policy management systems, can deliver these benefits across a range of insurance decisions. Its basic components—business rules management, predictive analytics and strategy optimization—can be



applied not only to POS decisions, but also to renewals, marketing (including cross-selling) and claims processing. Companies can start out in any of these areas with rules-driven automation to boost decision efficiency, consistency and market agility. Predictive analytics can then be introduced, offering increased precision to risk analysis and decisioning to lift results further. Strategy optimization can then be applied for even higher levels of performance and profitability.

7.2.1 New Focus for IT/IS: Revenue Generation

Information technology has brought significant efficiencies to underwriting, claims management and other fundamental insurance functions over the past decade, reducing cycle times and costs. Today the insurance industry is focused on revenue, and a new generation of technology solution—generally referred to as Decision Management—is making a direct impact on revenue generation.

This white paper explores how Decision Management impacts revenue: through real-time point-of-sale (POS) decisions and improved channel management; expanded risk tiering and innovative pricing; speedy cross-channel deployments of products addressing a wider range of markets than ever before; and data driven selection and targeting of new business and renewal prospects.

Decision Management solutions share common characteristics:

- *Front-to-back.* While the previous generation of insurance IT solutions was deployed primarily for back office processes and decisions, Decision Management is focused on bringing decision-making capabilities up to frontline functions, particularly POS. Using rules driven automation and intelligent user interfaces, Decision Management links channels and self-serve websites with the back office and, by improving the quality of inbound data, enables straight-through-processing.
- *Business managers in control.* Decision Management solutions enable business managers to create and modify the rules driving their processes and determining their decisions. With no programming required, changes can be made immediately, without consuming precious IT resources, and simultaneously updated across all channels.
- *Analytic insights.* Decision Management rules driven processes easily incorporate data driven analytics. Analytics increase precision in segmentation and decision making by providing clear forecasts of customer behavior. They can also optimize the decision strategies for how a company performs quotations, new business application processing, policy renewal and marketing. An optimized strategy strikes the best possible balance between risk and revenue objectives, given all constraints (resources, schedule, portfolio risk management, etc.) while taking into account market and economic uncertainties.
- *Fast, inexpensive development and maintenance.* Decision Management applications can be developed and deployed 4 to 10 times faster than traditional applications—at 25-80% less cost. This is largely because the rules driving decisioning processes no longer have to be laboriously translated into programming code. Developers can write rules using a business-like language. They can also get users involved by providing them with spreadsheet-like interfaces, tables, graphical trees and web forms that make it easy for nontechnical folks to create and modify rules. Predictive models and optimized decision strategies can be added to rules-driven processes without programming. And rules, models and decision strategies can all be reused for subsequent projects.

While the focus is on revenue generation, Decision Management solutions for insurance also lower costs. For one thing, by improving inbound data quality, they enable back office systems to operate at full tilt and reduce the need for costly manual review. For another, they can provide enhanced decisioning, callable as a service by legacy systems—improving a claims processing system's ability to detect fraud, for example, or to identify subrogation opportunities. A policy management system could better manage renewals, including the complexities of when to re-tier and re-price, while also identifying the best candidates for cross-selling. Enhanced decisioning could also enable the insurer to save



money by gearing renewal campaign effort and expense to the right level given the value of the customer relationship.

This paper explores the revenue benefits of front-end Decision Management solutions, while also pointing out opportunities for carriers to achieve additional cost savings and efficiencies. All the capabilities described are achievable today with the software and services that make up the FICO Decision Management solution suite.

7.2.2 Making Revenue-Generating Decisions at Point of Sale

Personal Lines Insurance Carriers are intent on increasing the flow of revenue through channels, and Decision Management provides the means. It makes POS systems really work—fast, uncomplicated interfaces for agents and other POS users combined with straight-through-processing to back office systems means that up to 90% of requests for quotations and new business applications can be handled automatically.

Solving the inbound data problem—

A major impediment to straight-through-processing has been the poor quality of inbound data. Frequent mistakes and omissions have caused requests for quotations and new business applications to be kicked out of automated flows and queued up for manual review—and that costs time and money.

Decision Management solutions solve the problem by validating data up-front to ensure it is complete, within valid ranges/types and consistent with the ACORD XML object model. Systems should also be able to validate against the carrier's own rules and should provide an easy means for nontechnical business managers to add XML extensions to the industry model.

When inbound data is impeccable, back office systems are able to return quotes in real time and, in many cases, perform underwriting in minutes. Customers often leave the agent's office with a policy number, binder and identification card in hand.

Making it easy to do business—

Business rules management (BRM), a key element of Decision Management, orchestrates POS processes so that they are efficient for both the user and carrier. BRM separates decision logic from application programming code. The business rules (policies, regulations, calculations, etc.) that drive decision making are therefore visible, manageable and editable by nontechnical business users.

At the POS, business rules provide the intelligence behind online forms that ask only relevant questions. Embedded conditional logic dynamically changes questions, fields and formulas based on the data being input and applicable state regulations and corporate policies. Business rules also access data as needed from internal and external sources to pre-populate form fields, minimizing the amount of input required from POS users.

Frontline interfaces should also accommodate users not working online by emailing forms as PDF files. Agents can then complete the forms offline, delegate the task to support staff or even print out hardcopy forms so they can be filled out by hand. Completed forms can then be emailed or faxed back to the carrier. The business rules management system should have the ability to turn faxed forms back into PDFs and also invoke conditional logic to automatically generate follow-up PDF forms with additional questions, where necessary, based on the initial submission.

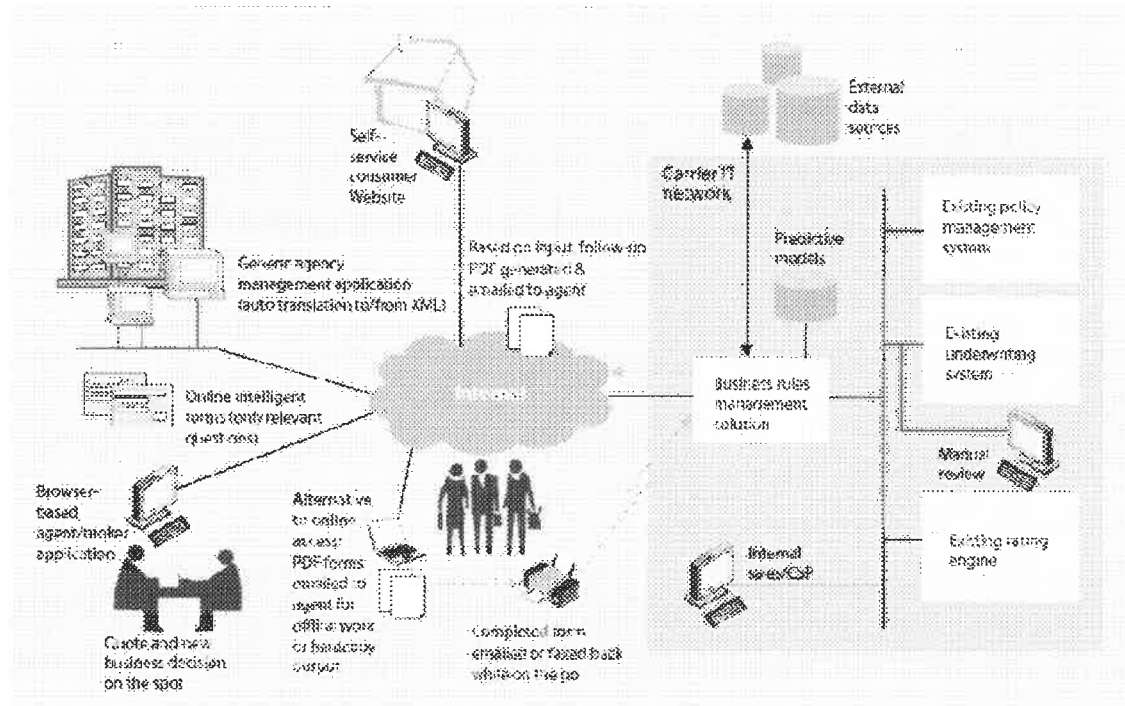


Figure 43—Straight-Through-Processing from Point of Sale

Increasing decision precision—

Predictive analytics can be easily added to rules-driven POS processes, helping existing underwriting systems make more accurate decisions. Many insurers, for example, are already including FICOTM Credit-Based Insurance Scores (sold through credit reporting agencies) as one of the many criteria used in their underwriting strategies. These scores predict insurance loss based on certain types of credit behavior—which have a proven statistical correlation with likelihood of claim submission. FICO also offers prepackaged and customized scores that directly predict loss ratio and loss ratio relativity—these scores are built on a broad foundation of insurance company data.

Risk scores can be used to make rules-based decisions about when to pull credit-based insurance scores from credit reporting agencies and access additional data sources—often the most expensive component in the underwriting process. For example, a company might create a rule that says it will not order MVRs or claims reports on new business applications with a score above a certain point. Below a certain point, it might automatically collect information from additional sources (loss history data firms, for example, collect information on property and automobile loss history), and route the application, based on the score and referral rules, to an underwriter with the appropriate skill and authority level.

Analytics, working in conjunction with up-front data validation, greatly reduces the number of applications referred for manual review. The added insight and clarity brought by additional data sources and predictive scoring also mean fewer applications fall into gray areas; most can be cleanly dispatched one way or the other.

For those that are referred, analytics increase the efficiency of manual review. Underwriters receive all the data and the scores, as well as codes and messages pointing out the reasons the application scored as it did, so they immediately focus their attention where it is required. Rules and scores may

also be used, in automated processes or guiding expert staff, for endorsement underwriting and to determine applicant eligibility for payment plans.

Improving performance at both ends—

While in some cases insurers are choosing to move their entire underwriting process onto a Decision Management platform (see case studies above), new solutions can also be used with existing back office underwriting systems. When deployed in this way, Decision Management becomes not only a source of agility at the POS; it's also a means of increasing the agility of legacy applications. Existing systems can call the new BRM-based decisioning processes and analytics as services—as shown in the following graphic. This approach extends the lifecycle and ROI from major investments while minimizing technical risk.

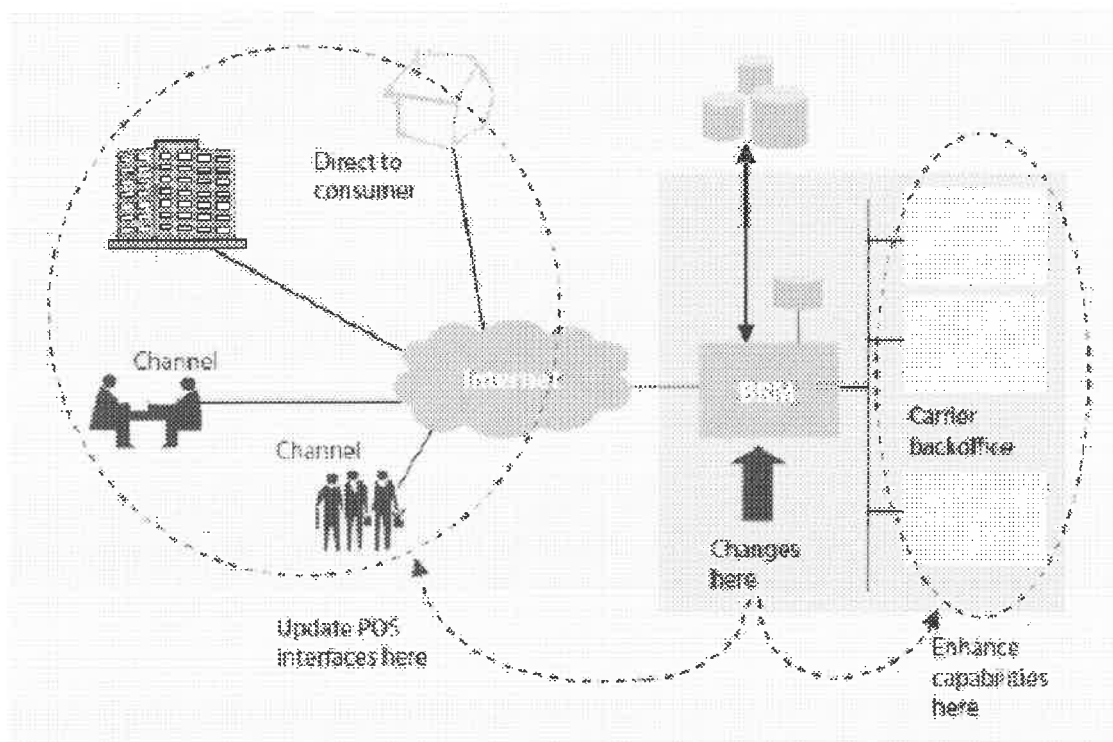


Figure 44—BRMS as a Source of Agility

7.2.3 Getting a Better Handle on the Channel

Channel management is greatly improved by eliminating the productivity losses and friction caused by constant calls and emails from carriers trying to track down missing information and correct mistakes. In addition, since underwriters are focused on fewer applications, those that truly require their expertise and discretion, they have more time. They can use it to make the right kinds of contacts with agents, brokers and other channel partners—contacts aimed at improving and nurturing valuable relationships. They also have some additional tools to do it with: a review of the scores derived from the applications submitted by an agency can assist in the evaluation of the quality of the business being submitted, and provide valuable insights into how the carrier can work with the agency to improve it.



7.2.4 Multiplying Profits and Markets through More Accurate, Innovative Pricing

Predictive analytics and rules are being used in conjunction with existing rating engines to improve pricing accuracy and flexibility. FICO's own experience in building analytic models for the insurance industry suggests that many insurers are "leaving money on the table" for their existing book of business through imprecise pricing. In addition, increased precision enables insurers to price coverage in ways that expand the range of risks they are willing to write, allowing increased market penetration and entry into new markets.

7.2.4.1 Using Real-Time Loss Forecasts to Drive Real-Time Rating

The very data that contributes the most value to a forecast of loss ratio relativity is data typically not found in industry rating methodologies. This data is among the key characteristics analyzed by FICO insurance risk scores however. Moreover, it is possible to combine risk scores with credit-based insurance scores and other types of characteristics to help create tiers that reflect not only risk but consumer behavior and price sensitivity also.

It should be noted that risk scores are not a methodology in themselves for creating multi-tier pricing structures nor a replacement for the actuarial methodology practiced by insurance companies and required by many states in rate filing. On the contrary, scorecards should be used to improve the precision of existing rating methodologies or to provide the pricing flexibility needed to accomplish strategic objectives for growth or market expansion.

Predictive modeling is a natural complement to rating processes, in fact. Scores are generally output in numeric rank order, which lends itself well to tier assignment. By setting granular score cutoffs and combining these with tiering rules, carriers can determine the number of tiers by product and refine tiering based on market and loss experience.

Below is a very simple hypothetical example of how score range could be divided into a six-tier price structure with implied loss ratio relativity, which could then be mapped to an existing or proposed tier structure.

Price tier	Selected score range	Loss ratio relativity	LR index relative to Tier 3	Proposed tier factor	% earned premium [#]
1	184 & up	0.73	0.87	0.87	11.9%
2	172 to 183	0.77	0.92	0.92	11.0%
3	160 to 171	0.91	1.00	1.00	11.0%
4	150 to 159	0.99	1.18	1.18	11.9%
5	141 to 149	1.07	1.27	1.28	13.0%
6	130 to 140	1.08	1.29	1.28	12.9%
7	115 to 129	1.19	1.42	1.40	12.9%
8	Below 115	1.34	1.60	1.60	11.7%

Figure 46—Example of Score-based Tiering

One of the advantages of using a predictive scorecard as part of a tiering algorithm is that the in-force book of business can be scored and the impact of the new rate program assessed. Scores can then be correlated to rate level, resulting in the net written premium impact—actual dollar value



increase/decrease. The book of business could also be analyzed and segmented to identify trends and determine their impact.

Here is a simple simulation of the benefits that could be attained from introducing a predictive model and expanded tier structure into a personal lines auto book of business. This table contains data representing a strategy to move from a three-product price structure (sub-standard, standard, preferred) to a one-product structure with six price tiers. This data was then applied to a development database of over 10 million records to compare the two strategies.

Premium distribution	Score cutoff	Premium factor
15%	200+	0.87
20%	<200	0.92
30%	<178	1.00
20%	<152	1.18
10%	<133	1.29
5%	<114	1.42

Figure 46—Example of a Tier-based Pricing Structure

The results are show in the table below:

Private passenger automobile strategy	Claims	Premium	Loss ratio	Expense ratio	Combined ratio
Three products / three prices	148,963,085	200,040,302	74.5%	27.00%	101.5%
Single product six-tier pricing structure	148,963,085	207,544,861	71.8%	26.02%	97.8%

Figure 47—Sample Simulation Process Results

For simulation purposes, it's assumed that many variables remain constant across the comparison. Yet among the factors likely to change are growth rates. New business booking rates and revenue would both be expected to rise given that the expanded tier strategy would give the insurer an automobile product with price points covering a larger segment of the population and geared to consumer willingness to pay.

7.2.4.2 Optimizing Decision Strategies

There is another type of analytics that can be applied to risk tiering and pricing, and which will produce even larger gains (typically 5 to 35% over predictive analytics alone). Decision analytics enables companies to leap over the months of testing and iterations typically required to approach optimal strategies—and get to optimal sooner.

Shorter time to optimal is a substantial advantage in a rapidly changing market. If it takes many months to refine a strategy, by the time optimal is reached, conditions may have already changed. Many companies are forever chasing optimal while operating at suboptimal levels. Decision analytics enables companies to get there right away and operate at a high level longer—reaping the rewards—before conditions change.

How does decision analytics work? A decision model is created by mapping the mathematical relationships between all the factors that go into a tiering decision aimed at an overall goal, such as maximizing profits. The model can incorporate any number of scores (credit, loss ratio relativity, response, retention, fraud, etc.) as input along with numerous other criteria. It finds the best possible tradeoffs between competing objectives (maximum revenue, minimum losses, etc.) within specified constraints (resources, schedules, regulatory and policy requirements, etc.), and can also build in allowances for ranges of market and economic uncertainty. Below is a simplified example of what a decision model for tiering might look like:

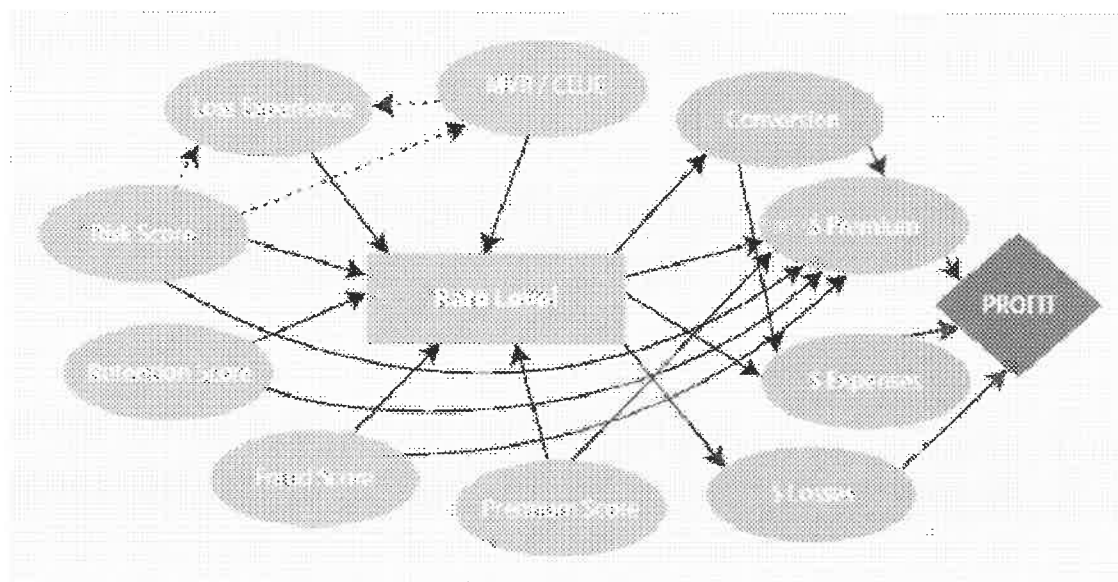


Figure 48—Sample Tiering Decision Model

The decision model is part of a multi-step process, which is summarized in the table below:

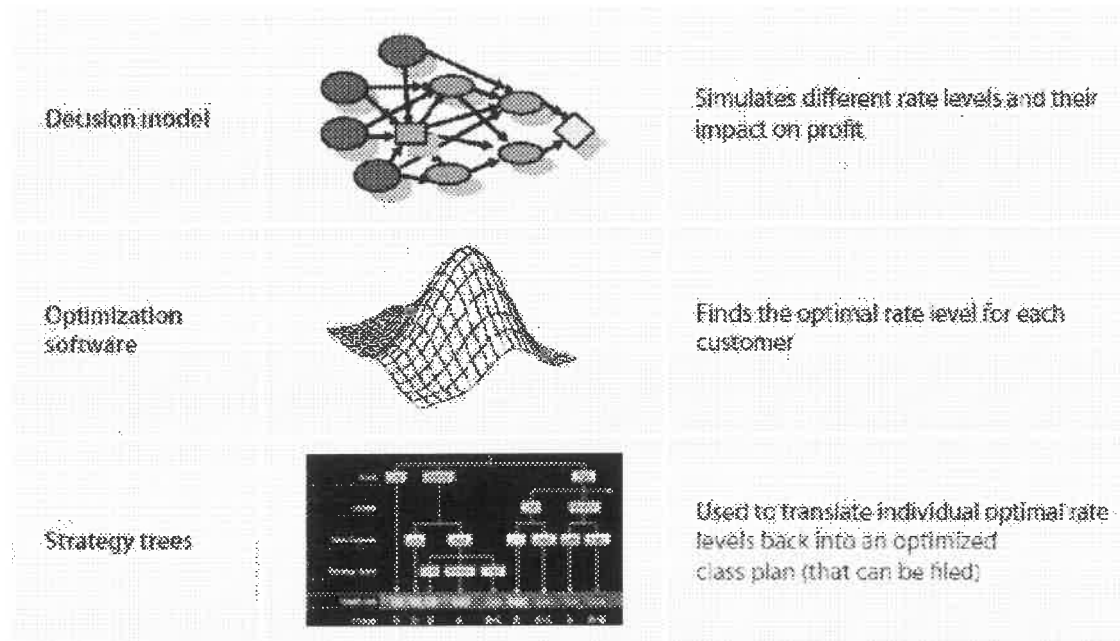


Figure 49—Multi-Step Decision Process

Strategy optimization not only lifts results, it also accelerates learning and the rate of continual improvement. New generation systems, for example, provide graphical tools (depicted in the third row of the table on previous page) that enable insurance analysts to explore how the optimal strategy is constructed, test the sensitivity of various "levers" on the performance objective, change variables, experiment with tradeoffs and ranges of uncertainty, and run numerous simulations very quickly to see alternative futures. Through this type of exploration, analysts can improve their understanding of and control over the pricing dynamics of their business.

7.2.5 Facilitating Profitable Renewals

Decision Management solutions can be used to segment a carrier's book of business with granularity that facilitates automated renewal processes and precisely targets policyholders with effective terms and packages. The results improve both sides of the balance sheet: higher renewal rates at less cost.

Renewal strategies, for example, can incorporate criteria based on account payment and claims records as well as various risk scores (credit, loss, and retention). Rules then automatically pull policyholders matching these criteria and assign them appropriate treatments before routing the renewal offer to the agent or broker. (Carriers may also save money by gearing the level of the renewal effort to the value of the customer relationship.)

As shown in the next graphic, rules can be used to make intelligent automated decisions about when to order external reports (when data is missing or has passed a specified shelf-life). They can manage the complexities of whether or not to re-tier (in states where allowed) and when to make a call to the rating engine (for surcharges, credits, etc.).



FICO Recommendations Whitepaper for Chubb Business Rule COE

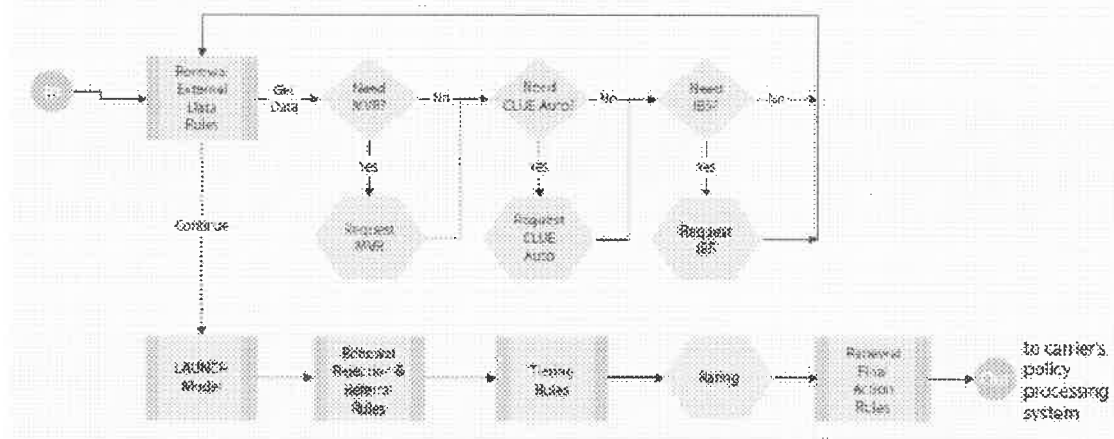


Figure 50—Sample Renewal Automation Flow

7.2.6 Expanding the Market by Tightening the Market

Increased segmentation, through analytics and rules, is also advantageous in marketing. Analytics provide insights that are enabling some insurers to extend into new markets previously regarded as too risky. Analytics can also be used in cross-selling existing policyholders and in new business acquisition.

A custom model can be created, for example, that enables carriers to identify the best candidates for life insurance among their auto and homeowner policyholders. Rules then distribute that information through the channel.

In new business, carriers can use analytics to make finer distinctions between market segments. Loss scores can reveal significant differences between applicants that appear to have similar risk or show hidden opportunities, such as willingness to pay more, that make additional risk worth taking on.

Analytics can be combined with experimental design and systematic internet-based pre-market testing to determine the most effective product feature sets and rate levels for the target segments. The result, as depicted below, is faster time to market for offers that have a high chance of succeeding at rollout.

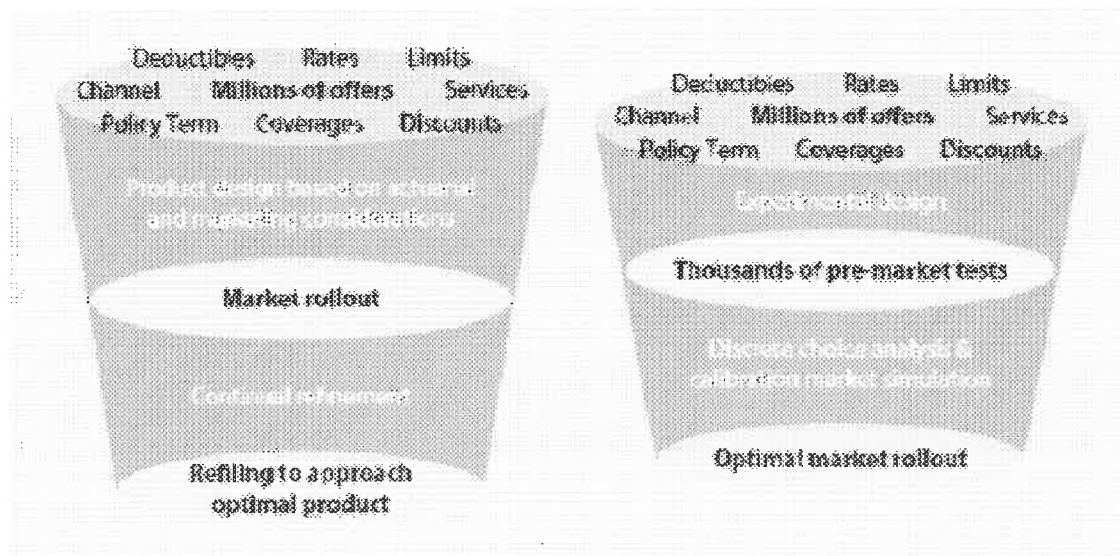


Figure 61—Analytics Fine-Tune Targeting

The Personal Lines Insurance industry has become highly efficient about processing claims in a timely manner. Adding flexible business rules management and powerful analytics to existing claims processing systems can increase speed and efficiency even further while reducing mistakes and costs.

Decision Management solutions, for example, can be used to create subsystems that work with claims edit applications. These subsystems can perform additional automated validation and ensure consistent application of company policies and state regulations. Rules can also guide claims processing specialists in handling exceptions according to best practices or in sensitive situations such as claims for death benefits.

Analytic models can be added to rules-driven processes to detect possible fraud, identify exceptional claims requiring special attention and spot opportunities for subrogation. For a workers' compensation carrier, for example, combining analytics with business rules management can boost the number of exceptional claims identified from the industry average of about 20% to up to 60%. In subrogation, predictive models can identify the 10-15% of claims with highest recovery potential within 24 hours of receiving accident reports. The models recognize even subtle subrogation signals that might otherwise be recognized late in the claim lifecycle, after a statutory filing deadline or perhaps never at all.

7.3 Decision Management Is the Next Step

The application of Decision Management to the insurance business is potentially quite broad—but it starts at the point of sale. Most carriers are intent on solving the inbound data problem because it affects nearly everything else. Once the quality of inbound data is reliable, carriers can boost automation levels, derive full benefit from back office systems and better utilize staff resources. At the same time, a straight-through-processing environment powered by flexible business rules and analytic intelligence permits highly competitive carriers to move beyond the status quo to innovative pricing, new markets and higher levels of profitability.

No matter where an insurance company currently is in its use of automation and analytics, Decision Management offers a natural next step. It builds on the efficiencies already gained in the back office,



enabling carriers to bring them forward to the point of sale for accurate, consistent realtime decisioning. Companies new to Decision Management may want to start with business rules management. Predictive analytics can then be added into these processes to lift results further. Decision analytics, incorporating all predictions and all other decision factors into an optimal strategy, offers even higher levels of performance and profitability.



8 Business Rule Lifecycle Management

This section summarizes FICO's recommendations for a Rule Maintenance & Versioning Strategy for Chubb Personal Insurance (CPI). These recommendations are the result of a series of interviews that FICO conducted with CPI architects and business leaders, by which FICO gained an understanding of the issues and concerns that CPI has about maintaining and versioning business rules.

This section also incorporates findings from similar sessions with Chubb Specialty Insurance (CSI). The result is a unified approach to business rule lifecycle management that should be broadly applicable throughout the various insurance companies within Chubb & Son Insurance.

8.1 Overview

As rules become more broadly adopted throughout Chubb, various application areas are beginning to look for standard mechanisms to ensure not only the quality, but also the validity, traceability/auditability, history and versioning for their business rules as they are maintained and extended over time.

Processes are already in place within Chubb for source management, versioning, verification, testing, quality assurance, and publication of traditional source code. Chubb recognizes that the artifacts in a BRMS repository are different from traditional source code and sees obvious gaps where their traditional processes will not be adequate for BRMS. New processes are needed for BRMS.

Bottom line: business users, technical users, and managers all want to apply formal processes to ensure that only "good stuff" goes into Production.

Key Drivers:

- Desire for an automated lifecycle processes
 - To keep track of rule changes
 - To also keep track of model deployments
 - Parallel rule authoring efforts
 - Feedback loop into development
 - To better prepare for the future
- Step in the direction of full enterprise decision management

Important to look at this as business rule lifecycle management (BRLM) and not simply source code version management.

This section discusses the following aspects of a program for managing the lifecycles of business rule repository content:

1. *Blaze Advisor Product Features and Proper BRMS Structure and Design*—there are many features within Blaze Advisor that will assist Chubb in managing the content of their business rule repositories. While some of these features are being used, their use is not consistent and globally applied. There are other features which Chubb is either unaware of or has chosen not to use. This section discusses these features at a high level.
2. *FICO's Best Practices for Rule Versioning*—Blaze Advisor has been designed to work out-of-the-box with several of the leading commercial off-the-shelf (COTS) version management systems (VMS) being used in the industry. Blaze also provides its own Blaze Versioning System (BVS) for customers who do not have a version management system or are using an



unsupported COTS VMS. This section discusses some FICO best practices for version management.

3. *FICO Best Practices for Business Rule Lifecycle Management (BRLM)*— This section provides an overview of FICO best practices for BRLM and use cases for typical BRLM scenarios. It is a predicate to the discussion of FICO's Lifecycle Management Service Framework discussed in Section 8.5.
4. *FICO Lifecycle Management Service Framework*— Lifecycle Management Service is a framework for managing the evolution of rules from creation to deployment in the Production environment. This service is delivered with a default implementation you can customize or replace with your own workflow tool.
5. *The "Maker – Checker" Approach*—FICO PS has developed an approach to managing some of the unfavorable side-effects from extensive use of management properties in large BRMS projects that are very volatile or undergo a high rate of maintenance. This section provides details on this approach.
6. *FICO Recommendations for CPI Rule Management*—this section provides a summary of the findings from the onsite sessions where FICO worked with Chubb CPI and CSI to understand Chubb's requirements and formulate suitable recommendations. It provides FICO's recommendation for an enterprise approach to Business Rule Lifecycle Management for the Chubb enterprise and then specific recommendations for CSI, CPI Print and CPI Underwriting.

8.2 Blaze Advisor Product Features and Proper BRMS Structure and Design

There are many features provided by Blaze Advisor to help manage the content of business rule repositories. There are also several design patterns developed by FICO PS. While some of these features and patterns are being used, their use is not consistent and globally applied. There are other features design patterns which Chubb is either unaware of or has chosen not to use. This section outlines FICO's recommended structure supporting Lifecycle Management through proper use of Blaze Advisor product features and proper BRMS structure and design.

This section is intentionally high-level and conceptual. The intent is to call attention to the product features/practices and tie the features to Business Rule Lifecycle Management. Details are available in the standard product and best practices documentation that FICO provides with Blaze Advisor.

8.2.1 Overview

There are dozens of product features and design patterns that facilitate management of the lifecycle for business rule repository entities. This section discusses the top 10 which provide the most benefit.

1. Blaze BRMS Component Structure
2. Blaze Repository Physical Structure
3. Blaze Repository Logical Structure
4. User Authentication and Authorization (A&A)
5. Lifecycle Management for Decision Service
6. Lifecycle Management
7. Filters
8. Management Properties (Repository Entity Metadata)
9. Difference Query

10. Graphical Difference Tool

8.2.2 Blaze BRMS Component Structure

Figure 52 shows the fundamental structure of a Blaze Advisor BRMS. Understanding this structure is key to properly implementing BRMS solutions. Every Blaze Advisor developer should commit this diagram to memory and be able to reproduce it.

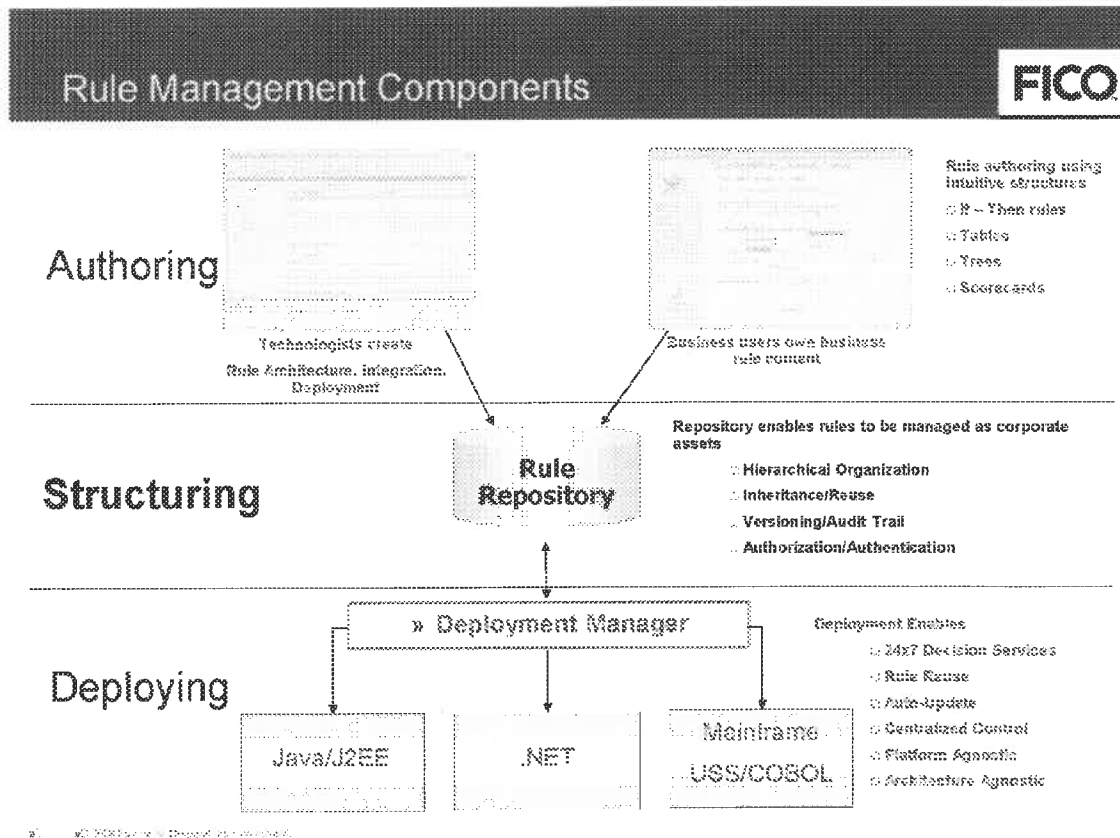


Figure 52—Blaze Advisor BRMS Component Structure

Key design patterns that precipitate from this view of a BRMS component architecture include the following:

1. Separate your requirements so that there are clear abstractions between development, maintenance, storage, and deployment. There should never be a reason to accommodate maintenance/extensibility implications because of the way something needs to be stored or deployed. The following are some exceptions:
 - a. The integrated development environment (IDE) and the rule maintenance application (RMA) may be slightly coupled. Certain aspects of design to facilitate maintenance and extension will often impact the way repository entities are implemented in the IDE.
 - b. The physical storage of the repository should be as abstracted from any IDE/RMA or deployment requirements as is possible. Some real-world exceptions are:



- i. The physical location of the repository and the hosting environment for the RMA can sometimes be coupled with application environmental requirements (i.e., firewalls) and multi-tenancy requirements.
 - ii. The format of the repository (RDBMS vs. file-based) can sometimes be coupled with deployment architecture and production environmental requirements.
2. Do not look at decision services like COBOL subprograms or like server-based applications (i.e., an RDBMS server, or a Websphere installation). Each decision service is a small stand-alone application component, a Java bean or .NET assembly. You can deploy as many decision services as you need within one application. Be as granular as possible. Fine-grain services are easier to maintain and extend than large monolithic services.
3. The inclusion of an RMA is indeed optional. However, deciding to forego an RMA does not give license to forego a good template-based design within the repository. FICO often hears people differentiate between "RMA Rules" and "SRL Rules." THERE IS NO SUCH DELINIATION. All rules are SRL rules, whether they are template-based or hard-coded. FICO best practices recommend implementing all rules as template-based rules regardless of whether they are intended to be edited via an RMA. This practices produces the most flexible, maintainable, and extensible BRMS.

Best Practice Update: Innovation

"Innovation" is a name for an older methodology of first implementing a decision service using hard-coded SRL for the purposes of quickly getting a service through testing, deployment, and into production with the intent to come back later and "innovate" or "templatize" the solution and creating a BRMS. The innovation approach to implementing a BRMS has long-standing track record for failure and is no longer recommended by FICO PS.

4. Do not forget about Deployment Manager. DM is there to help keep the rules in runtime working memory in sync with the rules stored on the file system.

8.2.3 Blaze Repository Physical Structure

Figure 53 show the physical structure for the repository recommended by FICO. It has four task-oriented sections, or folders.

1. *Technical Library*
 - o Components enable fast assembly of new services
 - o Templates enable RMAs and reuse
2. *Business Library*
 - o Rule content – policies, practices, procedures
 - o Efficient authoring via rule reuse
3. *Decision Services*
 - o Easy packaging/deployment of rules & analytics
4. *Testing*
 - o Organize business test cases

FICO recommends using the top-level folder names precisely as specified above. Using these exact names will clearly communicate your design to anyone familiar with FICO best practices. It will help any

experienced Blaze architect understand specifically what each folder contains and where to go to find whatever he may be looking for.

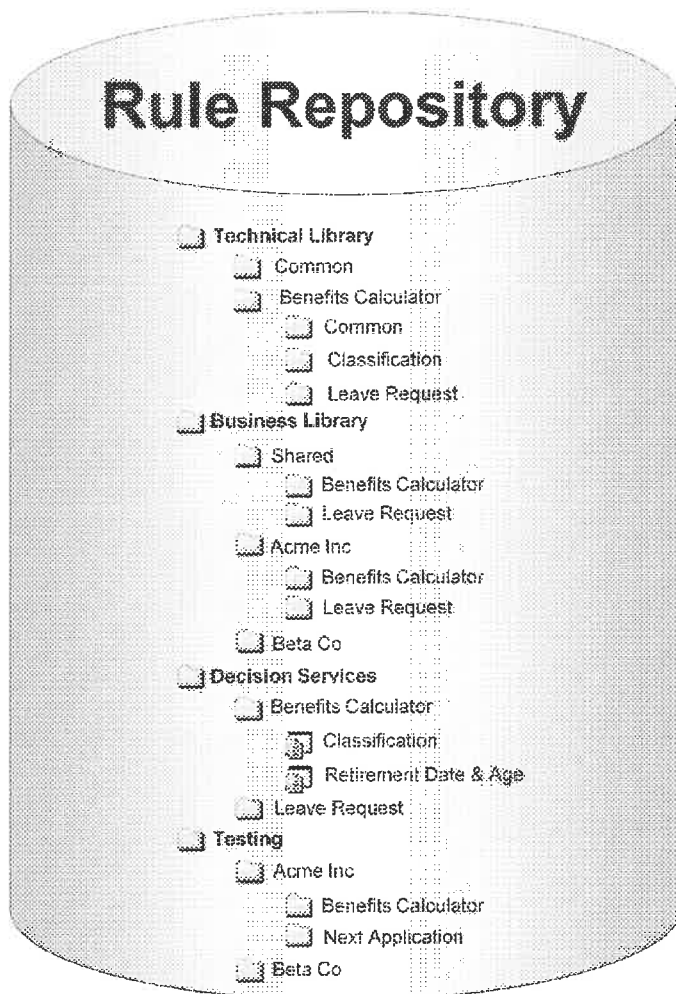


Figure 53—Recommended Repository Physical Structure

Separating the repository contents into folders like those discussed above will benefit business rule lifecycle management because it separates and abstracts various repository entities that are traditionally on different lifecycles and experience disparate rates and types of change.

The above physical folder structure is also important for the repository logical structure as described in the following subsection.

8.2.4 Blaze Repository Logical Structure

Figure 54, below, shows a logical project structure that allows for sharing, reuse, extensibility, etc. both within projects and across multiple projects.

Multiple Services include the same components

- Common components can be shared across services
- Referenced components reflect any changes/additions to the original rules
- Simplify maintenance by writing rules only once
- Reuse existing rules to quickly enter new lines of business

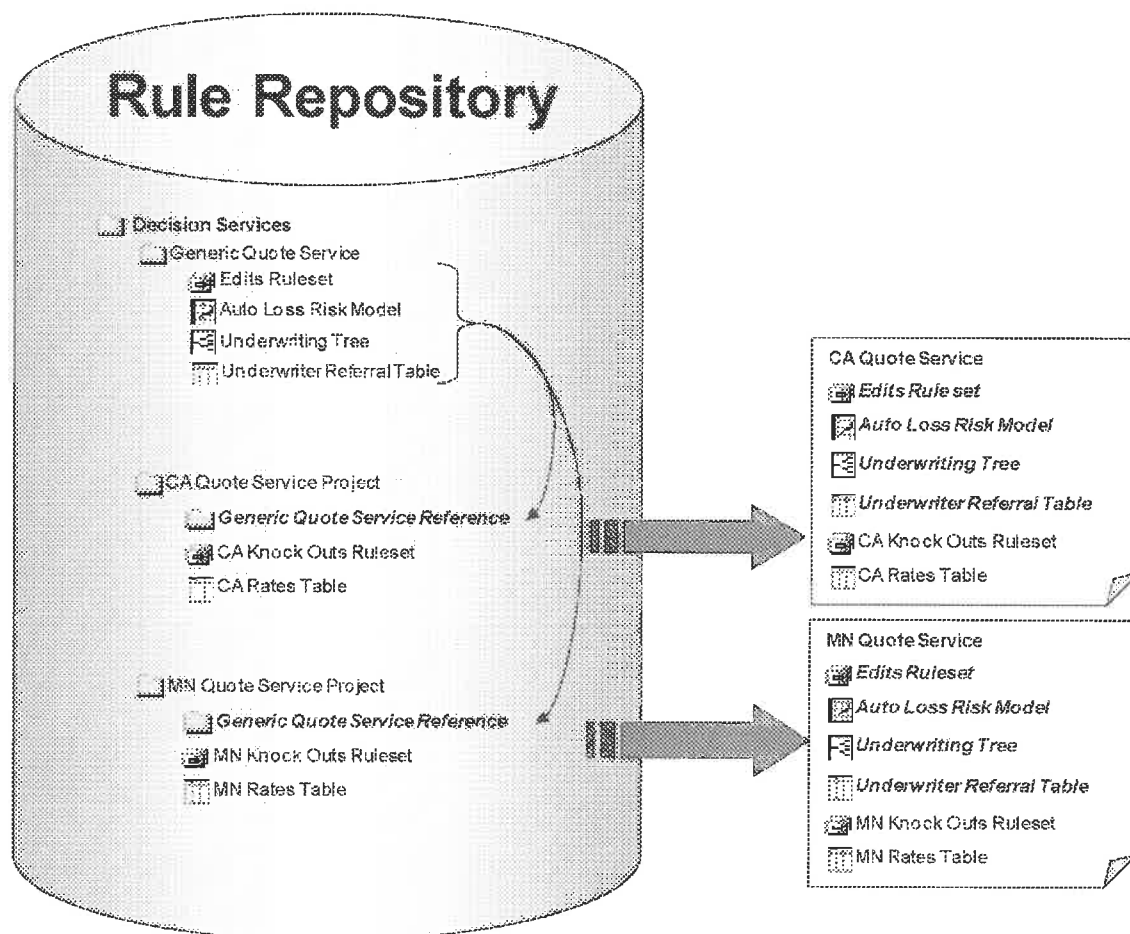


Figure 54—Logical Structure of Projects for Sharing and Reuse

The example in Figure 54 shows the deployment of two decision services for rating auto insurance policies. The insurance company has a set of common decisioning components that are shared across their markets and specific knock-out rules and rate tables for each specific market. By breaking out the common components and sharing them across deployments, they can be reused by reference wherever needed. And, most importantly from a maintenance and extension perspective, they can be isolated from the impact of changes within the various market-specific deployments.

8.2.5 User Authentication and Authorization (A&A)

User authentication and authorization (A&A) is an important supporting technology for rule maintenance. Blaze Advisor offers a number of features for authenticating IDE and RMA users and managing their rights and privileges to make changes to the repository.

- Provide access control for all repository components—Repository organization aligns roles with access
- Leverage existing systems for access control
- Control content authoring by using access control to drive templates

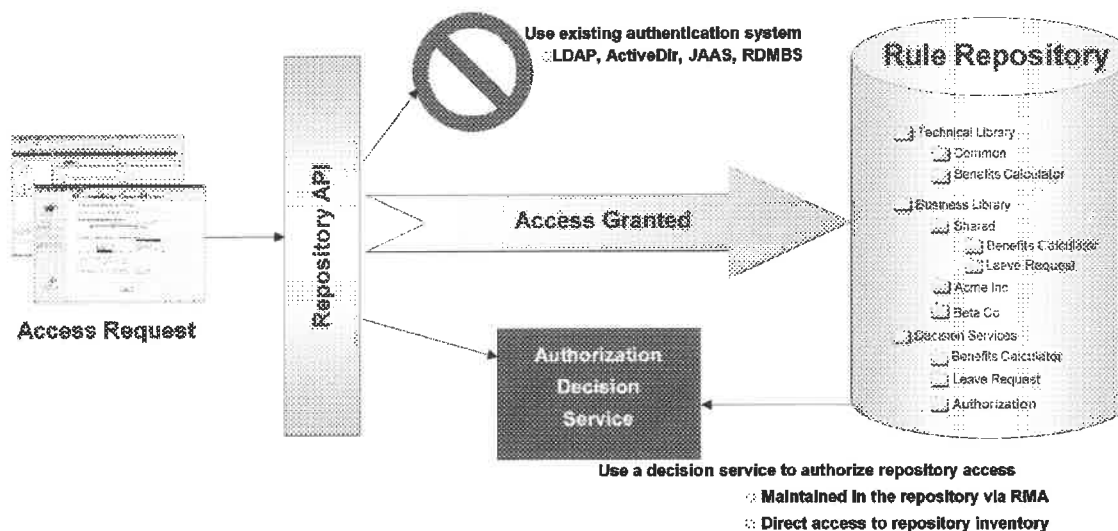


Figure 66—Example of RMA/IDE User Authentication and Authorization

8.2.5.1 User A&A Considerations

Key points for User A&A in Blaze include:

- Some basic functionality is provided out-of-the-box. However, most practical applications require customer implementations using the User A&A manager and the Blaze Advisor Repository API.
- The A&A manager does NOT provide file-system security. It controls access to the repository via the repository API and consequently the Blaze IDE and RMA. File system security is delegated to the environment.
- The API is open enough to allow customers to use their existing authentication systems (i.e., LDAP, ActiveDir, JAAS, RDBMS, etc.) or to create a custom authentication system specifically for the Blaze BRMS.
- Keeping the concepts of Authentication and Authorization separate and distinct from one another will greatly simplify the implementation.
 - Authentication is the act of confirming that a user is valid and is allowed to access the repository. It does NOT include any determination about rights or privileges.



- Authorization is the act of determining the rights and privileges. It is reasonable to presume that authentication has occurred prior to authorization. In this approach, authorization is typically a look-up to see what repository components the user is authorized to see and what his rights will be when he sees those components.
- Authorizations can be role-based or individually prescribed. Hybrids can be implemented using a role-based approach with individual rights granted to groups of only one individual.
- Authorization schemas do not need to be overly complex. Most BRMS applications require only rudimentary permissions:
 - Read
 - Update
 - Author
 - Approve
- In the above schema, the authorization are additive.
 - None (meaning no permissions are listed)—the user cannot see the entity
 - Read—the user can only look or reference at the entity
 - ... + Update—adds permission for the user to make changes to the entity
 - ... + Author—adds permission for the user can create new content in the repository
 - ... + Approve—adds permission for the user approve changes

Occasionally, a more complex schema is needed to handle specific application requirements. However, it is rare to see a schema with more than 6 or 8 different permissions.
- Many Blaze customers are incorporating the power of decisioning within their authorization process. Rather than simply looking up permissions for each component, it is often simpler to create rules about permissions. E.g.,
 - If the rule is an underwriting rule, and the user is the underwriting group, then the user can read it
 - If the rule is a rating rule, and the user is an actuary, then the user can update it
 - If the user is a manager, then the user can approve.

8.2.5.2 User A&A and Version Management

Proper user authentication and authorization is an important component of version management. Most version management platforms (BVS included) do not care whether a user was allowed to make a change. They simply manage the change. This out-of-the-box change-tracking is shown in Figure 56 below.

Standard change tracking capabilities are typically sufficient for most BRMS applications. Out-of-the-box, Blaze Versioning System allows you to:

- Show who made any specific change and when was the change checked in
- Show the history of all changes to a repository entity
- Show all the changes that a particular user made
- Include version numbers, last modification date, check-in date, status, comments, and user information with history information



However, some clients occasional have non-function requirements for auditing the change history on the repository and verifying that changes were made by authorized individuals. Providing this level of auditing requires an integration of the user A&A with the version management. It can be accomplished by either:

1. Extending the history information to include the credentials/permissions of the user
2. Integrating look-ups into authorization tables or queries on the authorization rules with the reporting on history.

This functionality is not available out-of-the-box, but can be implemented using the Blaze repository, A&A, and version-management API.

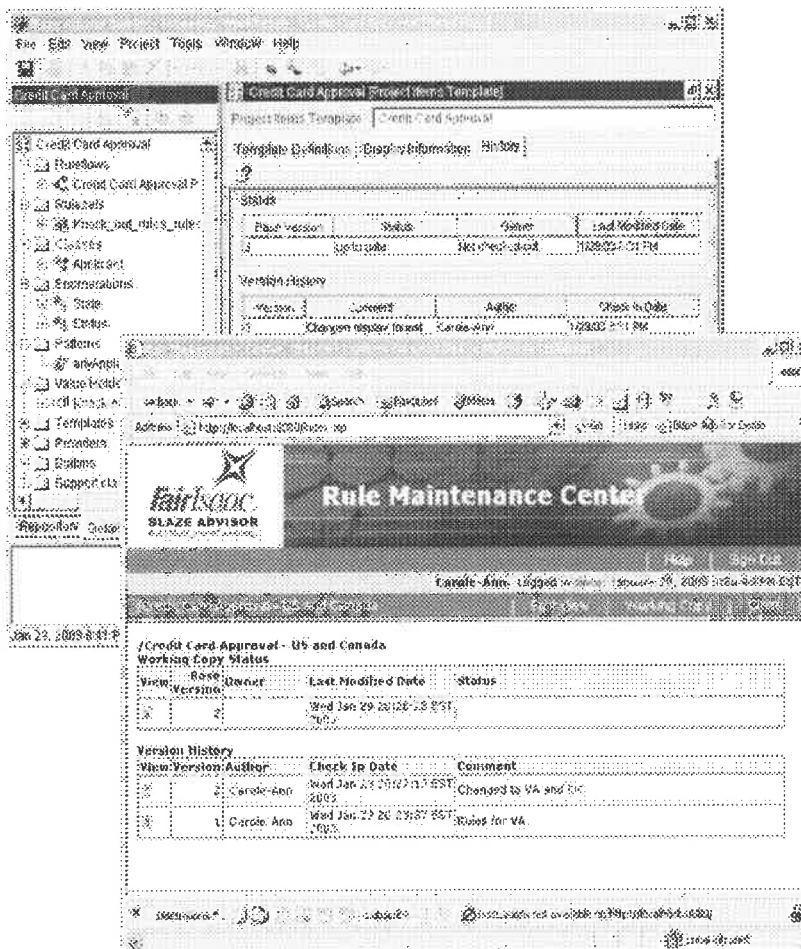


Figure 56—Version History in IDE and RMA

The discussions about BRLM to this point in this section have been mostly around using product features to appropriately structure the components and repository of a BRMS to minimize the scope of change management, and using A&A to manage the users who are making the changes. The

remaining subsections under this Section 8.2 will focus on product features practices for managing the actual lifecycle, and then on product feature for reporting and visualizing changes.

8.2.6 Lifecycle Management for Decision Service

Figure 57 below shows FICO's standard conceptual model for managing the lifecycle of decision services as the application matures and progresses through various environments. This discussion is intended to be conceptual and the model is intended to teach the concepts. This is not intended to be a design pattern or to be in any way applicable to real-world implementation. Real-world design patterns are included later in this document; see Section 8.4 below.

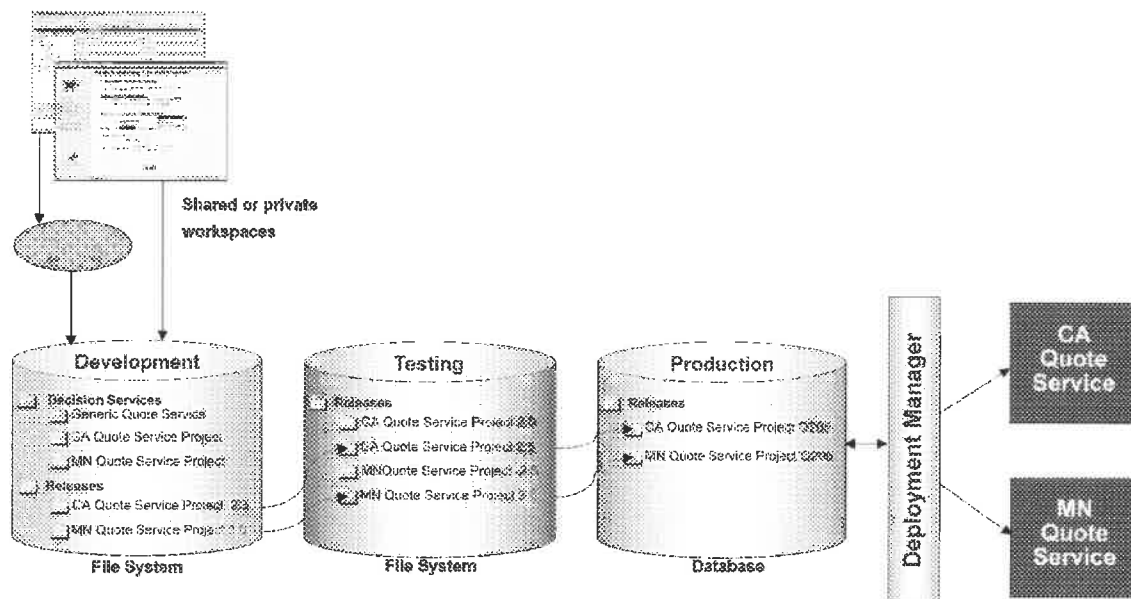


Figure 57—Lifecycle Management of Decision Services

In this example, we have two decision services: CA Quote Service and MN Quote Service. Each has their own rules and share common rules, as specified in the Decision Services folder in the Development repository. There are 3 repositories: one file-based repository each in the Development and Testing environments, and one database repository in the production environment. The two deployed decision services are being managed by Deployment Manager, which is monitoring the contents of the Production repository.

In the snapshot above, there are different versions of the two different decision services in each environment. For CA, version 2.0 is in production while version 2.5 is in test. For MN, version 2.0 is in production while version 2.5 is in test and version 3.0 is ready for release from development.

All editing is done in the development environment. The repositories in the staging and production environments are for runtime execution only within those environments. If a rule needs to be changed in production, the change is made in development and then pushed to testing. If testing discovers a bug, the fix is made in development and a new release pushed back to test.

This centralization of editing offers some key features:

- Developers use private workspaces as Sandboxes. This allows for free-form development (as is Agile) without concern for activities in the development environment interfering with repository content in staging or production environments.

- RMA users can use shared workspaces when collaboration is needed on large projects or projects with integrated workstreams.
- Package and Release projects for Decision Services and Libraries. Allows different versions of rules to be packaged with different deployed versions of the decision services.
- Flexible to fit any SDLC process—proven to work with traditional/cascade. RUP, Agile, and others
- Migration utilities insulate user from persistence mechanism. Rule editors (often business users) do not need to be concerned with persistence, staging, and migration.
- Use of Filters, Releasing, and Publishing mechanisms or API's allows IT to automate some or all of this process (if desired)

The example above shows deployments being managed within separate repositories in each separate environment. The same concept applies, just in a much simpler context, when using pre-compiled projects (.adb files) instead of repositories in the testing or production environments.

Next we will look at a more fine grain level to managing the lifecycle of entities within the repository.

8.2.7 Lifecycle Management for Repository Entities

Figure 58 below is a use case describing the requirements for a lifecycle management console from an action FICO customer.

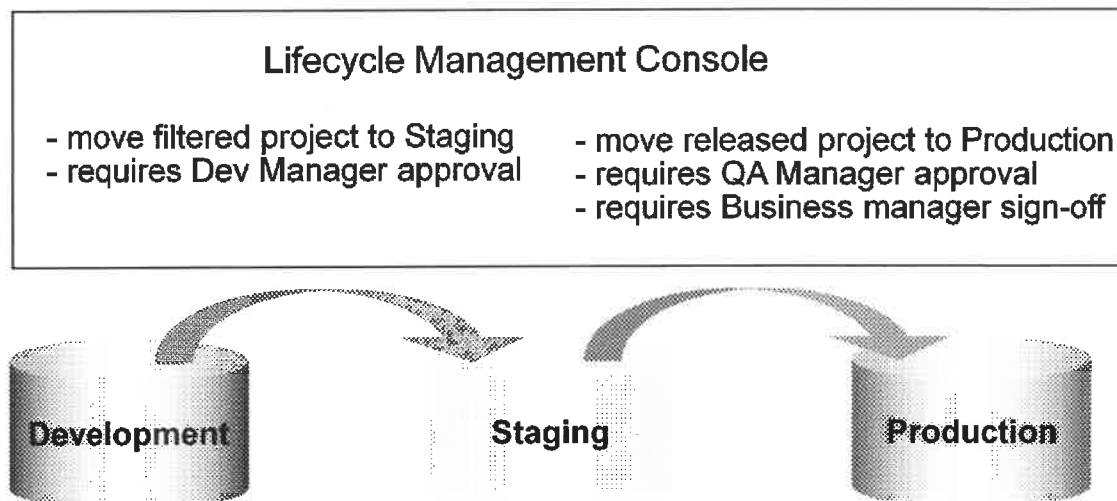


Figure 58—Lifecycle Management Use Case

This console was also required to automate the promotion of rules from one environment to the next and to incorporate a plug-in for the client's existing approval workflow system.

The following subsections describe the Blaze Advisor product features and practices used to implement the solution to this use case.

8.2.8 Filters

Blaze allows developers to define filters to sort through repository content. They are used to filter a smaller body of content from the larger body of repository content. This smaller body can then be

managed as appropriate (i.e., shown on an RMA page, deployed to a decision services, published to another repository, etc.).

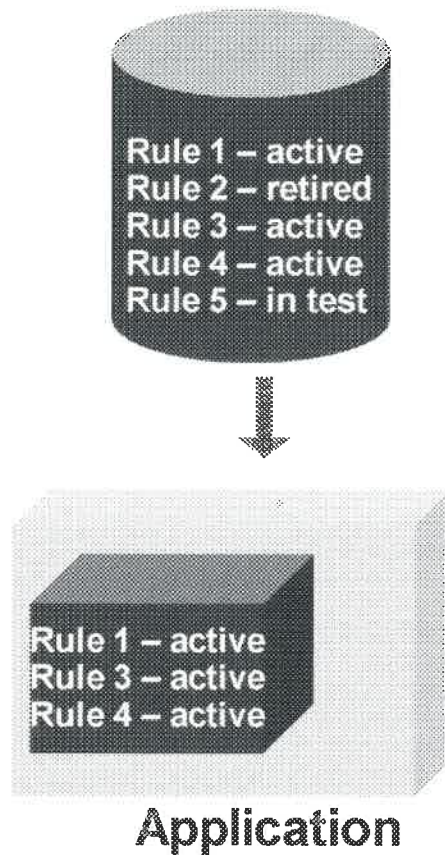


Figure 66—Example of Filtering Rules for Deployment

The example above represents a repository with 5 entities (in this case, rules). 3 of those rules are active, one is retired, and one is in testing. If you want the rule execution environment (application) to only use the 3 active rules, and to ignore the two rules that are not active, then filters can be useful.

Filters can be associated with

- Rules Server / Deployment Manager
- Release Management facility
- Etc.

Filters are based on

- Structural information: Name, type of entity...
- Versioning information: who modified what and when...
- Management Properties: entities with Status set to "Active"...

Filters can be applied programmatically and modified at runtime



- For redeployment using a different filter

The example above used a feature of Blaze Advisor called Management Properties. Coupling Filters with Management Properties is very powerful. The next section provides an overview of management properties.

8.2.9 Management Properties (Repository Entity Metadata)

The simplest way to think about management properties is “metadata for rules.” Management properties are equally applicable to any repository entity (i.e., class, named object, rule, ruleset, functional, decision metaphor (decision table, decision tree, and scorecard), ruleflow, etc.). Essentially, repository entity that is stored as an instance of a template can have management properties attached.

There are many uses for management properties. However, the most common are:

- Enable release management
- Make large repositories manageable

Some common examples are:

- Identifying the lifecycle status of rule versions, e.g., in development, in test, pending approval, stage, production, etc.
- Identifying the approval status of rule versions, e.g., approved for sys/integration testing, approved for staging, approved for production.
- Deprecating or “turning off” rules that are not longer in effect but need to stay in the repository for audit or post-date transactional purposes.
- Identifying the business unit that owns the rule, i.e., Personal Lines Underwriting, Commercial Lines Underwriting, Specialty Lines Rating, Corporate Premium Booking, and ET. Al.

One of the biggest cautions regarding best practices with management properties is to be judicious. Many Blaze customers really get their heads around this feature and get seduced into over-designing management capabilities in the repository. The risk is that the management properties themselves can become unmanageable. Some things to keep in mind:

- Do not reproduce information that is available through other mechanisms or features, i.e., versioning and audit information available from the version management system.
- Be careful about over-describing the lifecycle stages of repository entities. Considering (1) most SDLC processes incorporate 4-to-6 environments and (2) rules can have 5-to-8 lifecycle states, it is very easy to get caught up into describing lifecycle stages that are not really significant to the lifecycle management.
- Consider the value/returns on the granularity and frequency of saved metadata. i.e., do you need to record certain information on every edit, or just when edits are checked-in to the repository? i.e., what is the value of persisting fine-grain information on every rule in a ruleset? Would it be sufficient to store the information at the ruleset level instead?

And finally, do not use management properties for things that can be effectively captured in existing entity attributes. Some examples:

- If you have a well-thought naming convention, rule and ruleset names can be used to store a wealth of information for tracing implementation rules to decision requirements documentation.
- Every type of repository entity has a comment field. Use comments to save information that is relatively static and is not needed for any queries. Also, don't forget that the content of comments can be concatenated from valueholders. It is a good practice to use valueholders to generate comments that are well-structured and not dependent upon free-form entry by rule writes.



As a rule of thumb: Use management properties for metadata that you might need to use as keys for a query against the repository. Otherwise, look for alternative (version management, comments, etc.) mechanisms to store meta-information about repository entities.

8.2.9.1 Queries on Management Properties

Figure 60 below shows two metaphors for setting up repository queries in both the IDE and the RMA.

Using metaphors like these, it is very easy for rule developers and business analyst to search the repository using metadata.

Figure 60—Examples of Queries on Management Properties in IDE and RMA

Querying the repository on management properties is related to, but different from, and should not be confused with the Blaze Advisor product feature called a Difference Query. The Difference Query is another valuable tool to aid in Business Rule Lifecycle Management and is discussed in the next section.

8.2.10 Difference Query

Difference Query is a Blaze Advisor product feature that allows for a difference comparison between two projects, folders, or repository entities. This scope of this comparison is broader than standard queries. The Difference Query can be performed across repositories, which is very valuable to BRML when you have repositories in more than one lifecycle stage environment (i.e., DEV, TEST, STAGE, & PROD).



Difference Queries can be used:

- To perform a comparison between any two “things” within Blaze Advisor repositories. This includes projects, folders, and all of the many types of entities.
- To compare two things within the same repository or between two separate repositories
- With an optional filter to provide fine-grain precision in specifying what to compare

Difference Queries are defined in the IDE but can be executed and the results displayed in either the IDE or the RMA. Figure 61 below shows the IDE metaphor for setting up a difference query.

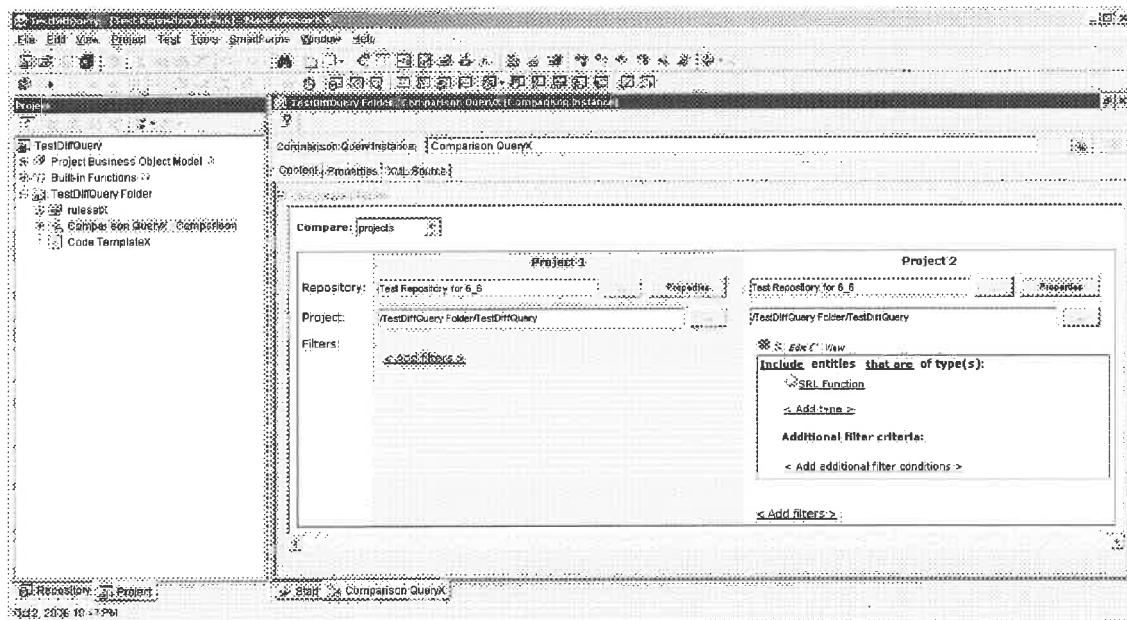


Figure 61—Example for Setting up a Difference Query between Two Projects

DQ returns the list of entities that are different, with meta-information on the differences.

- Collapsible/Expandable Result set view
- Structured expandable interface
- Color-coding for sources
- List of differences with human-readable details

Figure 62 below shows an example of a difference query results set being displayed in the IDE. A custom metaphor can be created to display the result set in the RMA if needed. However, typically, this level of analysis is not needed for rule entry.

FICO Recommendations Whitepaper for Chubb Business Rule COE

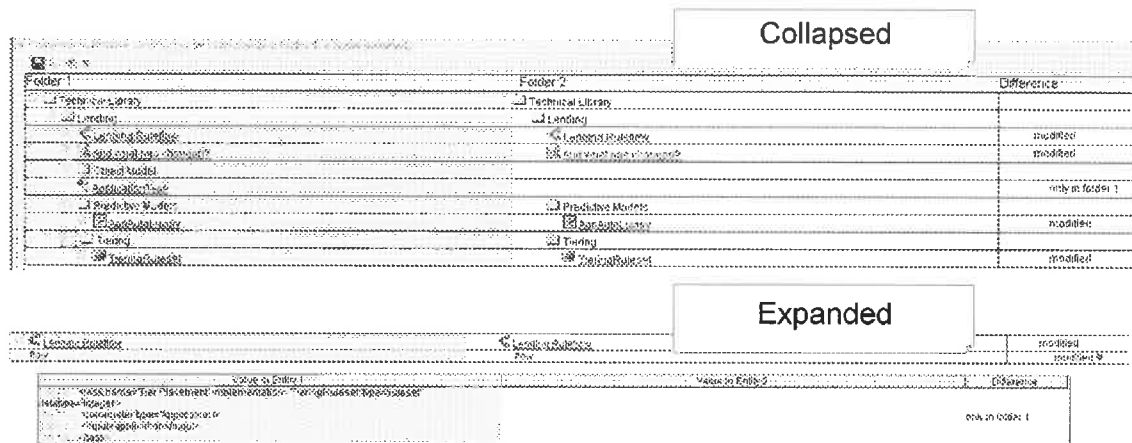


Figure 62—Example of Difference Query Results Set

Difference queries are very powerful and useful in BRLM. However, one drawback is that the output they produce is not in a great format for things like decision tables, decision trees, etc. In these cases, the graphical difference tool provides a much more intuitive metaphor for visualizing the differences.

8.2.11 Graphical Difference Tool

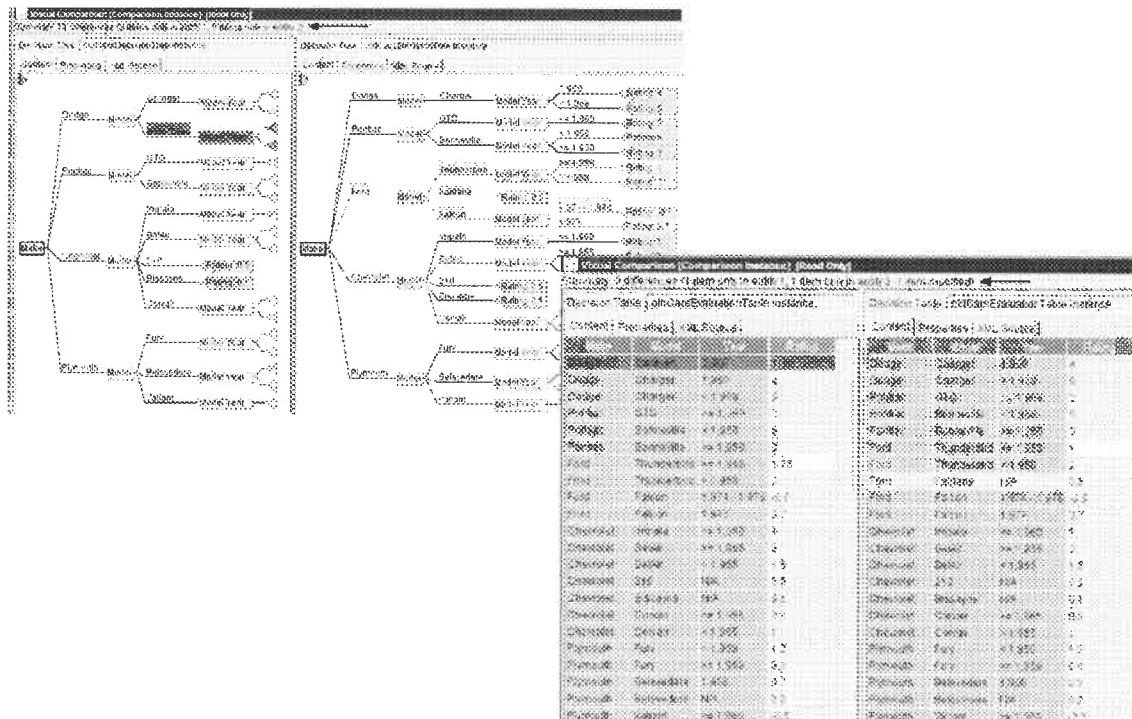


Figure 63—Examples of Graphical Difference Tool for a Decision Trees and Tables



The figure above shows the easy of visualization provided by the graphical difference tool. The red and green highlights make it quite easy to see that one branch was deleted from and another branch was added to the decision tree. For the table, similar color-coded fill colors are used to highlight rows that were added or deleted, or cell values that have changed.

8.3 FICO's Best Practices for Rule Versioning

Business rule lifecycle management (BRLM) and rule versioning are often used synonymously by non-technical contributors and managers. However, rule versioning is only one aspect of, or more appropriately, one tool used in, a larger program of BRLM.

Blaze Advisor has been designed to work out-of-the-box with several of the leading commercial off-the-shelf (COTS) version management systems (VMS) being used in the industry. Blaze also provides its own Blaze Versioning System (BVS) for customers who do not have a version management system or are using an unsupported COTS VMS. This section discusses some FICO best practices for version management.

There are a lot of concerns within Chubb about whether Blaze Advisor's version management capabilities are sufficient for managing the lifecycle of business rules within Chubb's enterprise. The frank answer to those concerns is: "Correct. Version management is not lifecycle management. So, Blaze's version management capabilities are not sufficient for BRLM." Version management is in fact a tool that is used as part of a business rule lifecycle management program.

The confusion, or misguided expectations, arises for a recent trend within the COTS VMS vendors over the past decade. Version management product features and capabilities fairly much stabilized around 1998. In order to compete and elude becoming commoditized, the COTS VMS vendors began to add application lifecycle management features to their VMS products. This trend has been going on since around 2000, and the result has been the modern-day availability of some very powerful VMS tools with very broad lifecycle management features. The net effect is that the line between version management and lifecycle management is blurred, arguably non-existent. Generally speaking, customers expect full lifecycle management features to be part of their VMS.

When it comes to using a COTS VMS to manage the versioning and lifecycle of a Blaze Advisor BRMS, there are two very substantial issues:

1. Blaze is not Java or C#. The tools are designed to manage the lifecycle of traditional application artifacts, work products, and outputs. There are not a lot of direct analogs between the files that make up a Java component and the entities that make up a Blaze project.
2. Blaze entities are much more fine grain. The lifecycle of Blaze BRMS entities needs to be managed at the instance level. COTS VMS products are designed to manage the lifecycles of application artifacts at the file level (.java, .class, etc.).

You need to have reasonable expectations over what can be managed by a COTS VMS. Just like you cannot expect a COTS VMS to manage the lifecycles of the individual methods within a Java class definition, you cannot expect a COTS VMS to manage the lifecycle of the individual rules within a Blaze ruleset.

8.3.1 Overview

This section discusses FICO best practices for version management using COTS VMS tools or BVS

1. Version Management System Considerations
2. Versioning Rules, Rulesets, and Decision Metaphors
3. Effective Dating



8.3.2 Version Management System Considerations

It is a FICO best practice to differentiate version management from lifecycle management. FICO recommends using either a COTS VMS or BVS to manage the following aspects of version-control. All other lifecycle-management requirements should be implemented using a special purpose tool or framework. Details on FICO's framework for lifecycle management are provided in Section 8.5 of this document.

The VMS should provide the following features:

- Version management at the instance level.
- Pessimistic locking of an instance upon check-out to prevent concurrent changes
- Check-in to release locks and capture audit information
- History on all changes made to repository contents
- Audit trails on all changes (user, date, comments)
- Roll-back to earlier versions without losing later versions
- Provide meta-data to Blaze filters, and thereby plug into the broader BRML program

FICO best practices are mostly agnostic with respect to the use of a COTS VMS or BVS. There are a few specific exceptions where one takes precedence over the other

- Use BVS to provide version control when the persistence mechanism format of the repository is LDAP or RDBMS. Neither of these platforms provide sufficient version management features.
- Use a third-party COTS VMS such as ClearCase or CVS to enable inclusion of external decision components from existing software or components on a separate life cycle processes

8.3.3 Versioning Rules, Rulesets, and Decision Metaphors

Understanding versioning for decision metaphors is often a challenge for people who have not yet adapted their thinking to a BRMS methodology. i.e., the similarity of a decision table to a spreadsheet or database table makes some people expect to see versioning at a row-level or even a cell level.

However, experience has show that fine grain versioning on decision metaphors is typically not necessary. In fact, a need for fine-grain versioning is more often a sign of a poor design than an actual requirement. This subsection will elaborate more on that concept.

8.3.3.1 Versioning Decision Metaphors

To understand versioning of decision metaphors, you must think about them as what they are: metaphors for displaying and editing the rules in a ruleset.

For example, consider the 9-rule ruleset represented in the decision table below.

Table 9—Sample Decision Table Metaphor

Gift Selection	Expensive	Moderate Price	Inexpensive
Adult	Wine Club Subscription	Gourmet Gift Basket	Bottle of Wine
Teen	iPhone	Fashion Clothing	Game Store Gift Card
Child	Playset	Doll or Action Figure	Pokemon Cards



The ruleset represented by this metaphor might look something like the code segment below:

```
ruleset GiftSelectionRules for {ageGroup: a string, budget: a string} returning a string
is {
    rule1 is
    if ageGroup = "Adult" and budget = "Expensive"
    then return "Wine Club Subscription";

    rule2 is
    if ageGroup = "Adult" and budget = "Moderate Price"
    then return "Gourmet Gift Basket";

    rule3 is
    if ageGroup = "Adult" and budget = "Inexpensive"
    then return "Bottle of Wine";

    rule4 is
    if ageGroup = "Teen" and budget = "Expensive"
    then return "iPhone";

    rule5 is
    if ageGroup = "Teen" and budget = "Moderate Price"
    then return "Fashion Clothing";

    rule6 is
    if ageGroup = "Teen" and budget = "Expensive"
    then return "Game Store Gift Card";

    rule7 is
    if ageGroup = "Child" and budget = "Inexpensive"
    then return "Playset";

    rule8 is
    if ageGroup = "Child" and budget = "Moderate Price"
    then return "Doll or Action Figure";

    rule9 is
    if ageGroup = "Child" and budget = "Inexpensive"
    then return "Pokemon Cards";
}
```

Changes to decision metaphors can take two basic forms:

1. Changes to the decisioning process
2. Changes to instances of the decision

8.3.3.2 Changes to the Decisioning Process

Changing the decisioning process for a decision metaphor means changing the way in which the metaphor reaches a decision. In the example above, this might mean adding a column for consideration of gender (male, female) in the gift selection process. This would have the consequence of doubling the number of rules and adding a 3rd condition to each.



Table 10—Sample Change to Decision Table Structure

Gift Selection	Expensive		Moderate Price		Inexpensive	
	Male	Female	Male	Female	Male	Female
Adult	Power Tool Set	Wine Club Subscription	Bottle of Scotch	Gourmet Gift Basket	Barbecue Gadget	Flowers
Teen	Xbox Game Console	iPhone	Xbox Game	Fashion Clothing	Game Store Gift Card	iTunes Gift Card
Child	Starwars Playset	Barbie Playset	Action Figure	Doll	Pokemon Cards	Pony Book

In this case the entire metaphor is changing, so versioning on the individual rules is irrelevant. All 9 rules need to be changed to incorporate consideration of either male or female and 9 additional rules need to be added to accommodate the alternative. In this case, it is much simpler and cleaner to version the entire decision metaphor.

8.3.3.3 Changes to Instances of the Decision

Changing an instance within a decision metaphor is changing the outcome of one scenario within the decisioning process. In the example above, this might be a change in the rule for selecting an expensive gift for a teenager (i.e., from an iPhone to an iPad) is not differentiable from a change in the overall gift selection decision.

Table 11—Sample Change to Instances within a Decision Metaphor

Gift Selection	Expensive	Moderate Price	Inexpensive
Adult	Wine Club Subscription	Gourmet Gift Basket	Bottle of Wine
Teen	iPad	Fashion Clothing	Game Store Gift Card
Child	Playset	Doll or Action Figure	Rubba Bandz Bracelets

While the 9 rules in the table above are indeed atomic and each does stand alone, they do not have a lot of business value outside the context of the ruleset. The version of the decision table is directly coupled to the version of the individual rules.

Further, even if a simultaneous change is made to the rule for selecting an inexpensive gift for a child (i.e., from Pokemon cards to Rubba Bandz bracelets); it is unlikely that such a change is occurring in a context that is unrelated to the first change.

A properly designed decision metaphor (ruleset, tree, table, or scorecard) should be focused on a single determination. It is rare for a properly design decision metaphor to be large or broad enough to encounter situations where simultaneous edits are made in unrelated context. i.e., it is unlikely to expect business group A to make the iPhone to iPad change simultaneously with business group B's decision to make the Pokemon to Rubba Bandz change.

8.3.4 Effective Dating

Effective dating should be considered at two levels.

1. Effective Dating on the rules
2. Effective Dating of a Metaphor

Effective dating is part of the rule entity. You can capture the start and end date of a given rule on the rule itself.

Effective Dating on the rules works against the system clock. If you need to have the rule compare against different date you need to either call the `clock().resetClock(myNewTime)` or have the rule do the comparison as part of the rule.

Adding effective dating at the table or tree level however is a bit more involved than going through the metaphor wizard.

You will have to do some group template implementation to wrap the decision table's instances. You can then also build rule template to fire the right version of the table or tree based on its effective dates. Below is an architecture approach that covers the versioning of tables and trees.

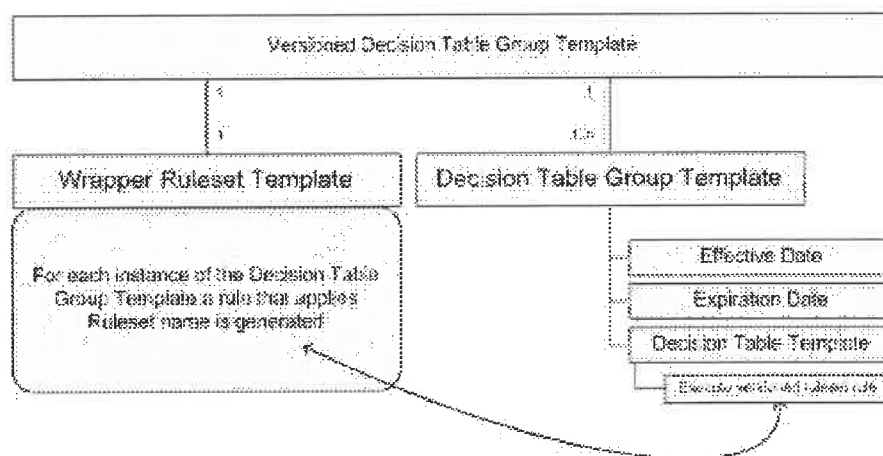


Figure 64—Adding Effective Dating to a Decision Metaphor

The image above describes a scenario that uses a Decision Table, but the same principles also apply to decision trees and scorecards.

It is known and understood that extensive use of effective dating can impact the execution performance of large rulesets. However, it is rarely necessary to effective-date every rule. Unless the application has extraordinary performance requirement and large rulesets, the performance characteristics of effective dating should not be a significant consideration.



8.4 FICO Best Practices for Business Rule Lifecycle Management (BRLM)

This section provides an overview of FICO best practices for BRLM and use cases for typical BRLM scenarios. It is a predicate to the discussion of FICO's Lifecycle Management Service Framework discussed in Section 8.5.

8.4.1 Overview

BRLM is FICO's proven practice of applying certain policies, processes and tools to the effectively give business users the power to manage business rules throughout its useful life

Our applications allows business users to define, manage and change the rules that guide decisions

Business users have the ability to change the rules as often and as fast as needed

Our BRLM applications are "decision neutral" – they can be setup for any kind of decision or process, from approving claims, credit initiation, to running a robot

BRLM includes all the phases of business rules from their beginning to their end. These phases include:

- Requirement analysis and rules harvesting
- Development and testing
- Deployment/Migration strategies

8.4.2 Blaze Advisor Product Features for BRLM

As discussed in earlier sections, the proper use of a number of Blaze Advisor product features can facilitate BRLM. The core of FICO's BRLM methodology is based mostly upon 4 Blaze Advisor product features.

1. RMA – Rule Maintenance Application
2. Versioning (BVS, CVS, SCM)
3. Management Properties
4. Release/Publish
 - Release can be used to tag a project with a release number and make a copy of the released project in another directory location in the same repository
 - Publish can be used to tag a project with a publication number and make a copy of the project in another directory location in a different repository

8.4.3 BRLM Migration Patterns

Business Rules tend to follow a different pattern of migration then regular applications

- Due to the expedited time to market
- Due to the different authoring environments

Unlike traditional application there tend to be multiple authoring environments

The following sections cover some common migration pattern we see with Business Rules

8.4.3.1 The Simple Push

The Simple Push pattern is the simplest BRLM pattern. In this pattern there is no business rule maintenance outside of the development environment. Rules are defined and developed in DEV.

Rules are maintained and extended in DEV. The only artifact pushed from one environment to the next is a pre-compiled Advisor project (.adb file).

Any changes that are identified in Q/A or TEST will be effected in DEV and a new deployment generated. In this use case, the repository is analogous to the application source code (i.e., Java file) and the .adb file is analogous to the compilation product (i.e., Class file) of a traditional software system.

This scenario is very popular with Blaze customers because it is very simple and easy to manage and does not sacrifice much in the way of maintainability or extensibility. The only draw-back is the inability to effect change in production quickly.

FICO typically recommends this scenario for a customer's first foray into BRMS. It is almost always sufficient for the maintenance requirements of a first release and the customer can embrace more complex scenarios as their maturity with BRMS increases.

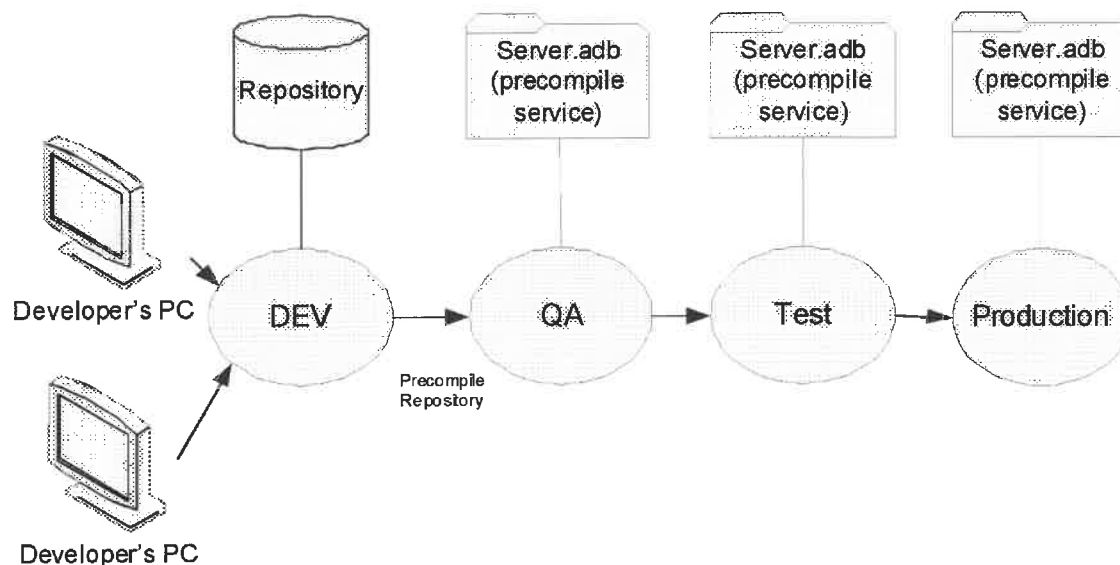


Figure 65—BRMS Scenario 1: The Simple Push

The scenario above would be contraindicated by any of the following requirements

1. Rapid change are required to rules in production
2. The SDLC cycle time is very short (i.e., Agile) and multiple versions of the deployed application may exist in each of the staged (meaning non-DEV, non-PROD) environments.

In any of the cases above, the next scenario may be satisfactory.

8.4.3.2 Maintenance in Staged Environments (Test or QA)

Maintenance in Staged Environments (typically TEST or QA) is by and far the most common approach to staging business rule repositories in the SDLC.

In this scenario, BRMS development starts out just like that described in Scenario 1. BRMS development and initial RMA work is performed in the DEV environment. However, once the BRMS is ready for promotion to DEV, the repository is pushed, published, or released to a staged environment. Most times this is staged environment is the application's normal TEST or QA environment. Some BRMS systems maintain an additional production-quality environment just for rules that is usable by the

testing and production systems. This allows production protocols to be followed to protect the repository, as it is indeed a valuable corporate asset.

Getting the business involved—this approach allows for both developer-centric and business-centric RMAs in the QA environment. Business users can help development by entering rules. They can review and approve rules entered by technical developers, they can execute tests and simulations to validate the decisioning.

Business rules maintenance occurs in an environment downstream from Dev. Therefore, any changes required for deployments in TEST or PROD will be effected in the QA repository and then manually resynchronized with the DEV repository.

Scenario 2: RMA maintenance in Staging only

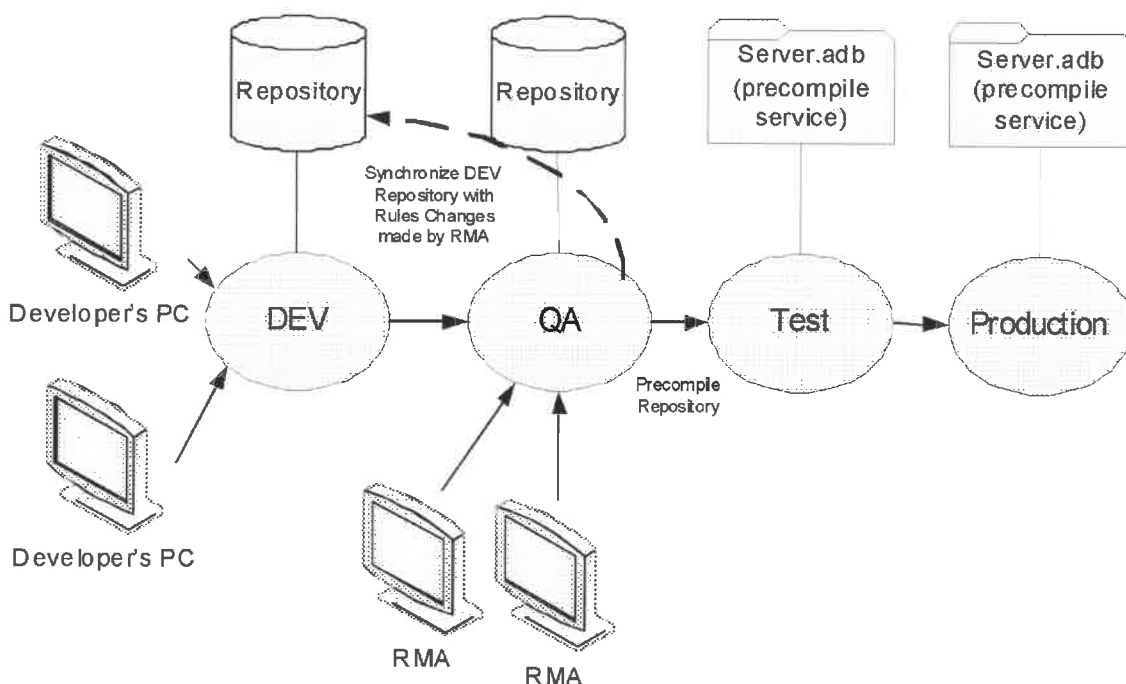


Figure 66—BRLM Scenario 2: Maintenance in a Staged Environment

8.4.3.3 Maintenance in Production

The final scenario is Scenario 3—Maintenance in Production. Many BRMS applications use business rules technology to facilitate rapid deployment of changes to business decisioning near real-time in a 24-by-7 operation.

24/7 Operation is really only supported with any degree of effectiveness by having a repository and RMA in production.

The scenario below plays out just like scenario 2 until the BRMS is promoted into production. Once the BRMS is promoted into production business rule maintenance on production-ready decisioning happens in the in the production repository using the production RMA.

Scenario 3: RMA changes to Production

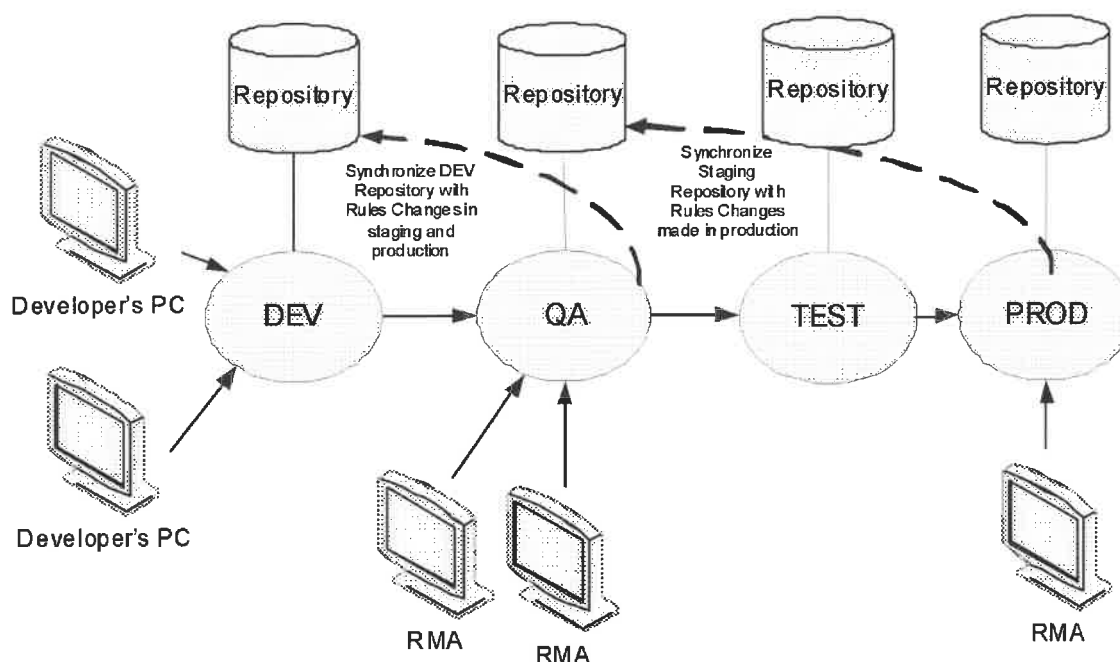


Figure 67—BRML Scenario 3: Maintenance in Production

To be effective, this approach requires testing capabilities to be provided within the production RMA (at a minimum). The business user making any change to production rules must run an exhaustive unit test suite as well and comprehensive regression test suite before releasing the changes. Once the changes are released, their effect will be immediate.

Additionally, this approach will realize strong benefits from incorporating decision simulation capabilities in the production RMA. This will allow the business analyst to analyze the impact (not just the correctness, but the business impact) of the change before releasing.

8.4.4 BRML Use Case

The following is a case study in BRML from an actual FICO customer.

Client requirements were

- Apply changes to production fast
- No protracted testing cycles
- Little as possible IT involvement
 - Automatic migration of new rules to production

- No interruption of services
- Manager approval process
 - Automatic notification of changes
 - Visual approval process
 - Automatic promotion to production (at rule level)

8.4.4.1 Challenges and Solutions

The following were specific challenges and the solutions devised by FICO PS

1. **Challenge:** Different changes present different risk—certain parameter changes presented serious risk to backend processes by potentially overloading certain databases

Solution: Quantify risk and move maintenance of risky rules downstream

2. **Challenge:** Due to two environments for changing rules; how to keep track of approval requirements in two environments?

Solution: Keep state of approval on the entity itself as management properties

3. **Challenge:** How to keep the two different environments in synch

Solution: Architect the solution so that there is no physical file conflicts between the different entities enabling the migration of files with no possibility of conflicts.

8.4.4.2 Environments

The following figure illustrates the environments for the use case. Note, developer PC's are in pink, business user's PC's are in blue.

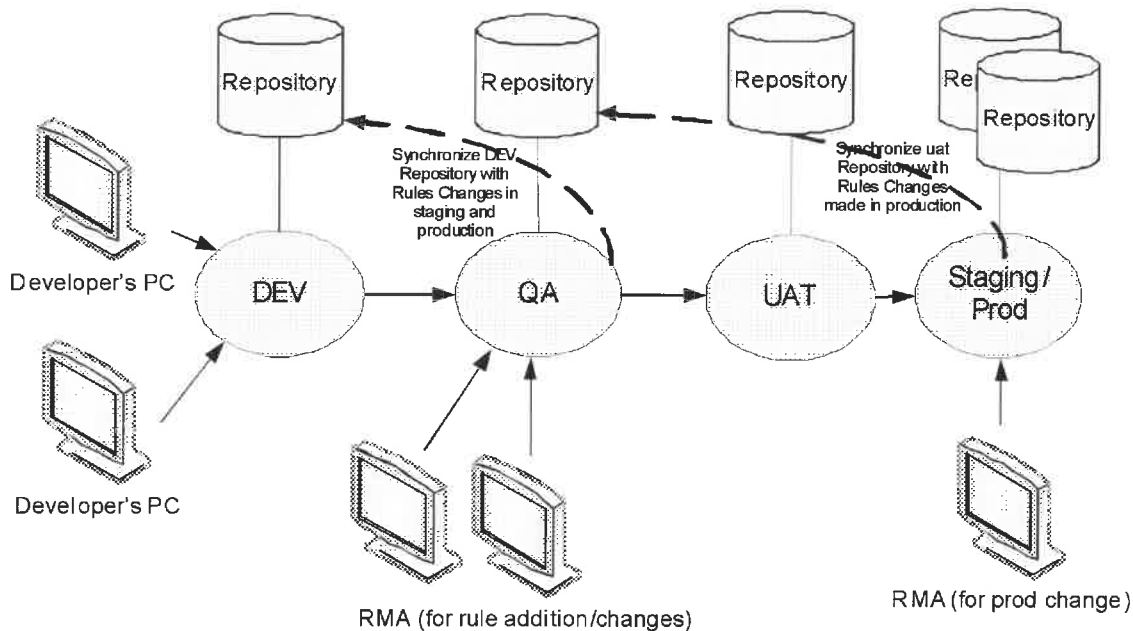


Figure 68—Environments for BRLM Use Case Example



8.4.4.3 IT Owns Changes to Repository in DEV

As illustrated in Figure 68 above, IT (developers) are responsible for all changes to the repository in the DEV environment. These changes would occur under Technical Library, Rule Services and Testing.

Promoting to the next environment would be promoting the entire repository (easiest solution).

Before promoting from DEV:

1. Synchronize repositories
2. Unit test should be run against newly incorporated Business Library

At this point the Repository needs to be migrated to QA.

1. QA RMA will be frozen at this point
2. QA will run the test cases it can (based on available data)
3. If issues arise feedback is given to IT (see flowchart)
4. If no issues arise the modified repository is migrated to UAT to run regression test cases

8.4.4.4 Business Owns Changes to Repository in Q/A

As also illustrated in Figure 68 above, all changes to the repository in QA will occur through the RMA. Both business and IT may make changes (IT being involved for more brUnit test definition than for business rule content). The Business is responsible for all changes to the business rules in the repository.

These changes would occur under Business Library only. In those instance files/folder that have been identified as user type of changes (as opposed to production type of changes).

Promoting to the next environment would be promoting the entire Business Library to UAT.

Before promoting from QA:

1. All changes from PROD environment (Business Library\Prod changes) need to be incorporated in the repository
2. During this time STAGING (RMA) needs to be frozen
3. QA needs to test the rule changes (RMA in RMA is frozen)
4. If issues arise QA is notified RMA is unfrozen in USER/QAT and fixes applied

At this point the Repository needs to be migrated to UAT.

- If issues arise feedback is given to USER

8.4.4.5 Static Repository in TEST

The customer's testing environment was essentially as static staging prior to production. No repository content changes were permitted in this staged environment. Functional testing was completed in QA. No changes were expected in TEST because the tests were focused around Systems, Integration, Performance, Environment, etc. If any repository changes were ever to be identified in TEST, those changes would be pushed back to QA or DEV as appropriate.

8.4.4.6 Business Owns Changes to Repository in PROD

First assumption is that changes in Production are a separate set of possible changes from changes in QA. It is extremely complicated to do parallel development against the same rules in two different environments. Once changes are made in the Staging RMA, first run battery of unit tests (BrUnit which can be run from RMA). Manager is automatically notified of changes. Manager can approve using an



approval queue. Once approved the changed instance file(s) (specific to production changes) are promoted into production. These changes also need to be migrated back to QA.

8.4.4.7 Additional Design Considerations

- A repository architecture that allows for moving granular components
- A repository that physically separates the code maintained by IT from that maintained by the business
- Approval tracking on those entities that require approval
- Automated notification of changes that require approval
- Automated migration/promotion of these entities that have been approved
- A deployment manager that will automatically refresh the rules that are executing in production

8.4.5 Real-world Considerations—Balancing Governance and Business Agility

Yogi Berra, the renowned manager of the New York Yankees once said,

"In theory, there is no difference between theory and practice. In practice, there is."

Unfortunately, the real-world does not allow us to solve complicated problems with theoretical designs. We need to make considerations for the way things actually work in practice. The following are some considerations that may need to be made in adapting the BRLM approach to the real-world.

8.4.5.1 Disparity in Business and IT Vision for BRLM

Generally, business wants BRLM process:

- to enhance business agility via shorter SDLC
- to reduce IT dependency for business rules changes
- to support approval, governance and compliance requirements

Business does not want BRLM process to increase operational risk

Generally, IT wants BRLM process:

- to focus on the types of changes that BRLM is properly designed for and reserve proper role for traditional SDLC
- to be implemented gradually with proper IT involvement
- to include proper check and balance steps, such as automated business logic verification and regression testing
- to include strategies for handling worst case scenarios such as production code rollback

IT does not want BRLM process

- to increase operational risk
- to encourage business to pursue big-bang fast track approach

8.4.5.2 Industry Perception of BRLM

Generally smaller companies with limited IT resources tend to embrace BRLM more eagerly. Larger companies especially those in financial industries tend to proceed more cautiously due to more complex



compliance and governance requirements. Every company should evaluate BRLM per its own business need, IT environment and existing SDLC process to develop suitable BRLM process.

8.4.6 Impact of SDLC on BRLM

BRLM only pertains to the life cycle of business rules, which is one component in an application, generally shorter than SDLC. SDLC governs all components in an application, including data components, integration components, etc.

If the required changes in an application only involves business rules but no other components, then BRLM can be followed and achieve faster speed to market. If the required changes involve other components, then regular SDLC needs to apply. Otherwise new business rules put into production per BRLM will not work properly or work at all.

Software industry recognizes the need to improve SDLC and such improvements have cut the lag between BRLM and SDLC.

8.4.6.1 The Agile Movement

The Agile movement started largely out of necessity by looking at ways to increasing efficiency by reexamining the way people communicate and create software. The Agile model was explained as a collection of simple principles and practices:

- One of the Agile practices is Continuous Integration (CI), the practice of executing series of build steps in a scheduled or automated fashion. The build scripts may be run by a computer or a human; they can be automated or manual.
- CI is an important practice for the Agile team. The CI system can provide feedback to all parties immediately after a problem is discovered.
- BRLM can be integrated into the CI process and tools such as Maven, Cruise Control, ant to achieve overall greater speed to market for different software component changes not just rules
- BRLM, when integrated into CI, reduce risk of deployment issues due to incompatibility between rules governed by BRLM and its dependent components such as object models governed by regular SDLC
- BRLM, when integrated into CI, reduce risk of deployment issues due to incompatibility between rules based decision services and software components that depend on such services
- Although Agile, and CI have make it more feasible to integrate BRLM as part of SDLC, business may still push for even shorter cycle than what SDLC can offer with Agile and CI.
- Business sometimes pushes for instantaneous rules change deployment due to various business reasons, which cannot be accomplished easily by SDLC based on agile and CI.
- The key is to have business measure the risk of such instantaneous changes against the benefits and design a plan to mitigate the risk.
- As part of the risk mitigation plan, CI can still be leveraged for supporting a mechanism that can roll back such instantaneous rules changes.

Longer term, SDLC will continue to build on the momentum based on the increasing popularity of Agile and CI, and evolve the processes and enabling technologies.

The speed to market and agility offered by BRLM may increasingly become the norm rather than exception in SDLC.

BRLM may leverage more and more the evolving process and technology infrastructure of agile and CI based SDLC and result in growing convergence of BRLM with SDLC.



Until SDLC is completely up to speed with BRLM, there is always a need to study the special process, and technologies associated with BRLM.

8.4.7 Summary

Business Rule Lifecycle Management

- Permitting business users to make act and make changes fast
- Minimal IT intervention
- Shorter cycles
- FICO applications allow business users to define, manage and change the rules that guide decisions
- Balancing Good SDLC Governance with Business Agility
- Minimizing Impact of SDLC on BRLM

8.5 FICO Lifecycle Management Service Framework

Lifecycle Management Service is a framework for managing the evolution of rules from creation to deployment in the Production environment. This service is delivered with a default implementation you can customize or replace with your own workflow tool.

This section provides a high-level description of how to set up and use the FICO framework. Detailed specifications are included with FICO's standard product documentation.

This discussion is geared towards readers familiar with Blaze Advisor as well as SmartForms for Blaze Advisor.

8.5.1 Overview

Organizations generally rely on established processes to enforce best practices around the creation, testing and publication of business rules. Those best practices are key to smooth and successful operations, especially as business rules spread beyond the first project. Experienced organizations often institutionalize those processes from the very first project. Roles such as rules architects, business analyst and business deciders are clearly identified and the responsibilities assigned. Automated testing and approval workflows are deployed to reflect those roles and responsibilities.

The following two roles are typically identified, at a minimum:

- *Rules Architect*: responsible for the implementation of technical rules and business rules templates into the Blaze Advisor development environment, often responsible for the generation of the appropriate deployment
- *Business Analyst*: responsible for the creation and maintenance of business rules through specifically tailored business analyst user interfaces (RMAs).

Along the same lines, business rules are subject to a predefined lifecycle which may vary per organization. Here is an example of those stages:

1. Development: business rules, templates and the overall rule flows are being developed, using the Blaze Advisor IDE
2. Technical testing: business rules are verified and tested for technical robustness and consistency, using tools such as the Verification service in the Blaze Advisor IDE, or brUnit test suites

3. Business testing: business rules are managed by business analysts and tested for business consistency and overall business performance
4. Performance testing: business rules go through performance and stress testing
5. Production: business rules are deployed in the context of an application, powering its decisions

Blaze Advisor allows business users to create or modify business rules in the Rules Maintenance Application as a result of fast-paced business condition changes and to push them into the running system via the Deployment Manager.

However, more sophisticated lifecycle workflows and controls are required to address the needs of managing deep changes to business rules and templates. Blaze Advisor provides a complete programming interface allowing the implementation of such workflows and controls, including but not restricted to the following operations:

- Execution of verification service using the powerful technology introduced in this release, including introspection of results
- Execution of brUnit test cases, test suites and implicit project-test suites, introspection of results
- Release of projects using optional filters within a given repository
- Publication of projects using optional filters from one repository to another (regardless of the technology)

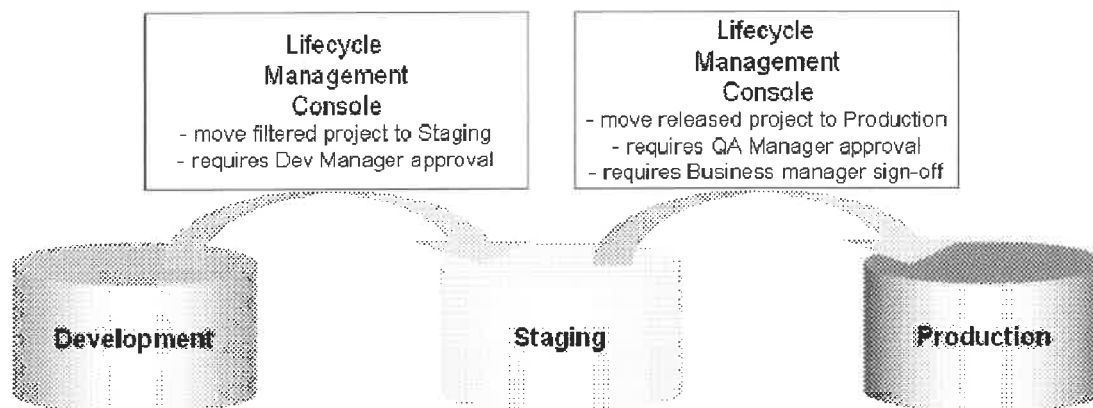


Figure 63—Lifecycle Management

Lifecycle Management refers to the consistent management of Blaze Advisor rule projects from initial Development through various Testing/QA processes to final deployment in the Production environment.

The Lifecycle Management Service addresses the physical migration of rules from one environment to the next. It is flexible enough to allow for customized processes that embody your methodology, with no assumption on which methodology you use. The Lifecycle Management Service can be deployed in conjunction with other tools that contribute to the overall business rules management.

The Lifecycle Management support in Blaze Advisor provides:

- Lifecycle management operations such as filtering, releasing, publishing projects

- Lifecycle management APIs to allow the implementation of lifecycle management workflows
- A Lifecycle Management Workflow reference implementation

The Lifecycle Management Workflow reference implementation included with Blaze Advisor provides an illustration on how to implement a configurable lifecycle process, allowing the implementation of controls to perform lifecycle management operations.

This default implementation is designed to illustrate how those interfaces can be leveraged in order to implement a workflow that controls and guides the management of the lifecycle of Blaze Advisor projects using those APIs. The reference implementation of the Lifecycle Management Service can be customized or replaced with other workflow tools.

Note that the reference implementation of the Lifecycle Management Service requires both Blaze Advisor as well as SmartForms for Blaze Advisor to be installed and enabled, and all relevant .jar files to be available in the CLASSPATH.

8.5.1.1 Lifecycle Management Workflow

The "Lifecycle Management Workflow reference implementation" is implemented using Blaze Advisor and SmartForms, leveraging the Lifecycle Management API. Support libraries implemented in `com.blazesoft.devtools.extensions.lcycle`

The repository for the default implementation can be found at:

`<ADVISOR_HOME>\examples\lifecycleWorkstation`

The two principal projects in this example are: "Services" and "Workstation"

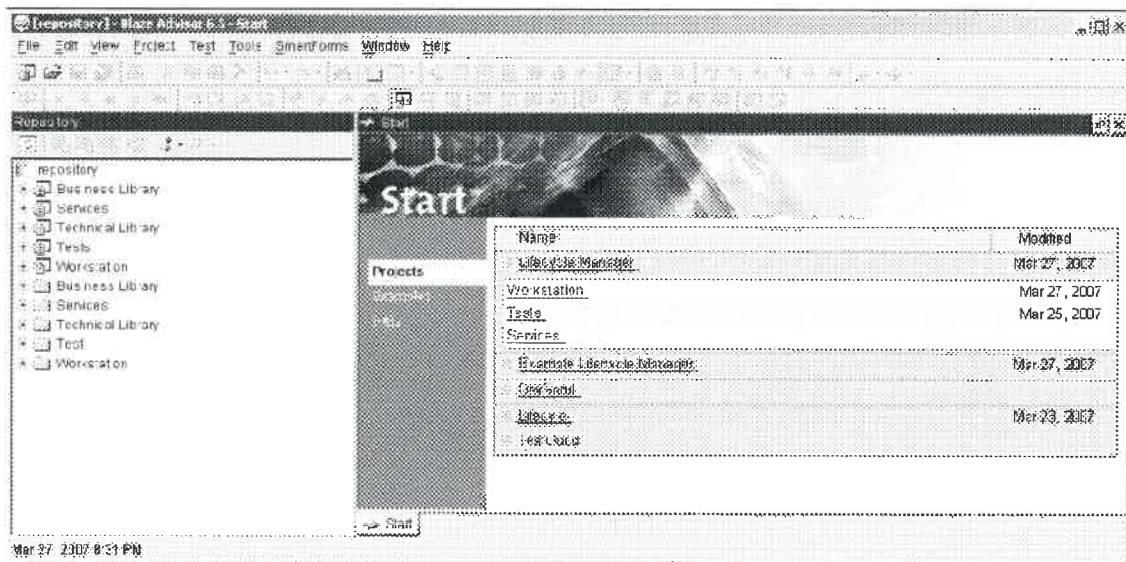


Figure 70—Services and Workstation Projects under FICO Lifecycle Manager

8.5.1.2 Lifecycle Management Services

The “Services” project contains an event-based ruleflow which implements the logic to process Lifecycle Management service requests and a set of functions, provided in the “Business Library”, which invoke the lifecycle API to do their work.

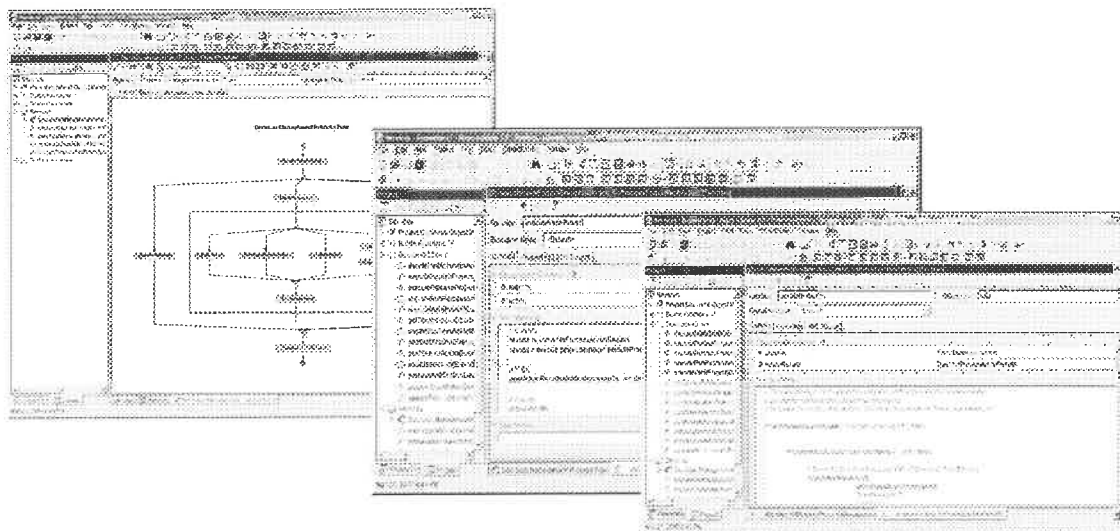


Figure 71—Contents of the Services Project

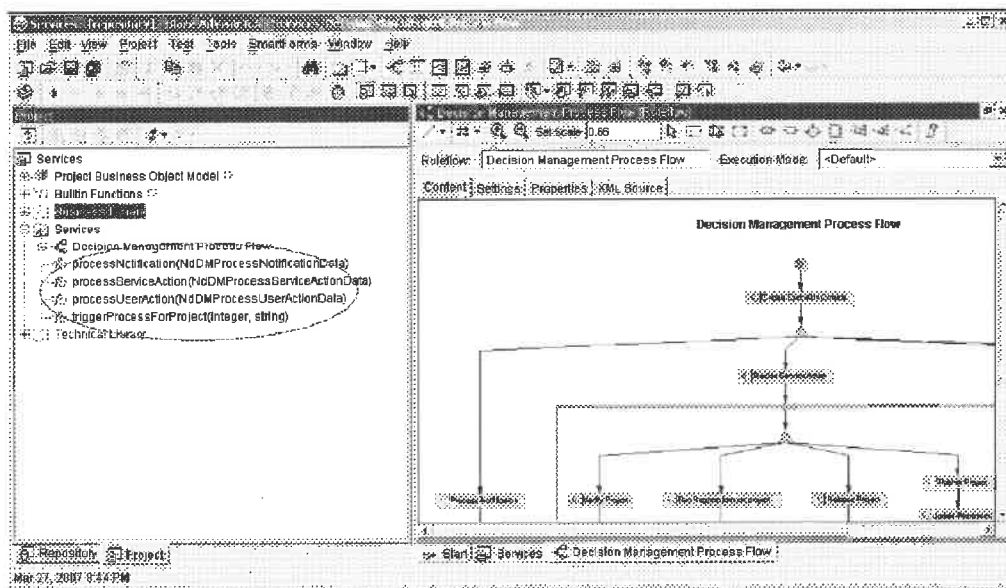


Figure 72—Services Project Ruleflow

The ruleflow and functions can be customized by technical users

In general, you are responsible for the deployment of this service and its integration with an environment that generates Lifecycle Management requests and feeds them to the service.

The service is deployed by creating a standard Blaze Advisor rule service that exposes one or more of the functions defined in the "Services" project as entry points.

For convenience the "services/" folder in the example includes a Blaze Advisor server deployment descriptor which refers to an example of a service deployed as a simple Java class.

The provided default deployment of the Lifecycle Management Service is implemented as a JavaBeans deployment

- Excepts a JDBC-ODBC access to the lifecycle management database through the jdbc:odbc:BlazeLifeCycle data source
- Monitors the state of the "service requests" queue
- Dequeues requests, and invokes the corresponding entry point
- Execution of the rule service results in
 - Actions derived from requests executed
 - Additional user or service requests queued
 - Notifications posted

To trigger the service, with the full "dev" classpath set,

- Make sure the database is set up
- `java com.blazesoft.devtools.extensions.lcycle.services.NdProcessLifecycleService`

Note that if this service is not running, the overall workflow will not progress

8.5.1.3 Lifecycle Management Workstation (or Console)

The "Workstation" project (a.k.a. the Lifecycle Management Console) provides the implementation of the Lifecycle Management Workstation which comprises several web forms - Login, User Action, and User Action Selection - and a SmartForms WebFlow which orchestrates the forms and applies back end logic to control the process.

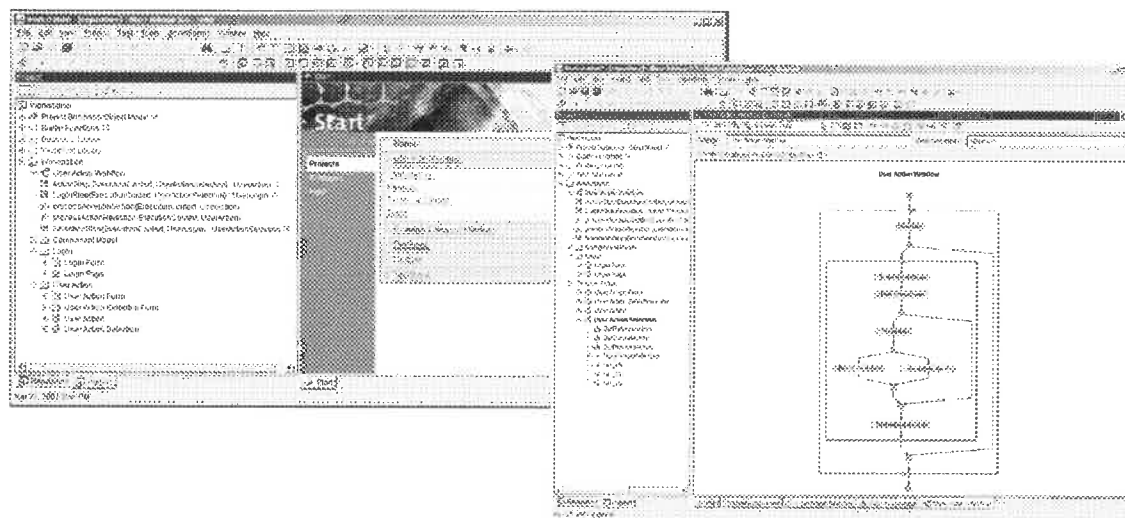


Figure 73— Contents of the Workstation Project

Technical users can customize the WebFlow and Forms as necessary

Note: the Lifecycle Management Workstation requires the SmartForms for Blaze Advisor add-on to be installed and enabled.

Note that the SmartForms implementation takes advantage of the “forms rules” capabilities of the tool to simplify its implementation.

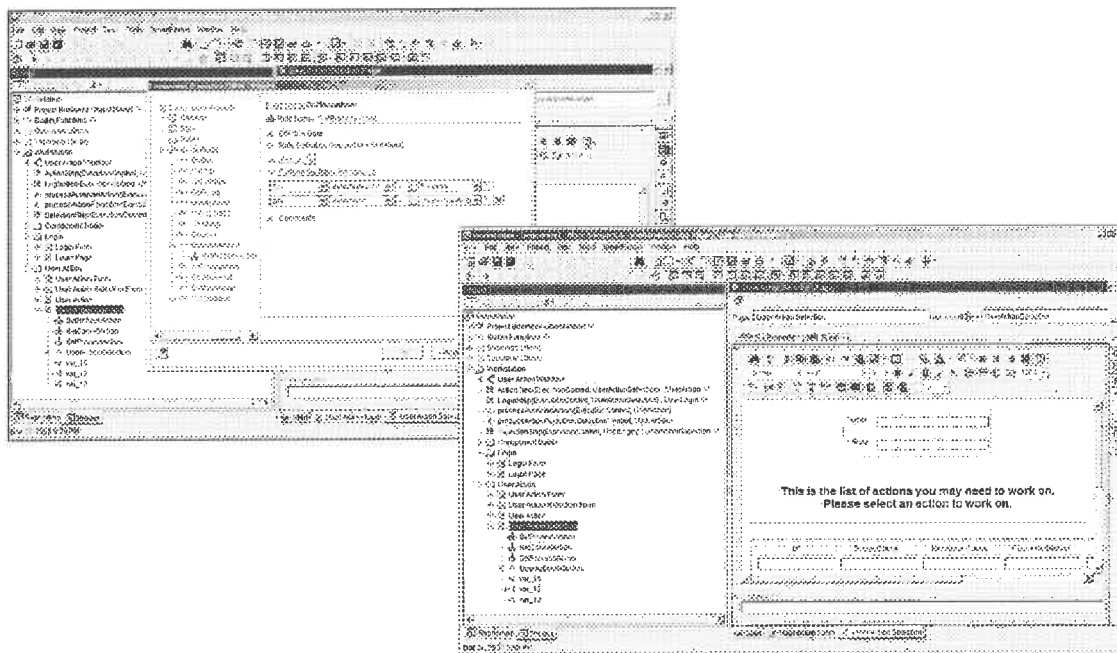


Figure 74—Form Rules Configuration of the Workstation

The WebFlow orchestrates the work for the different roles around the lifecycle management of the rules. The WebFlow can be deployed like any standard SmartForms for Blaze Advisor WebFlow. The Blaze Advisor "jars", including the development ones, need to be accessible to the deployment.

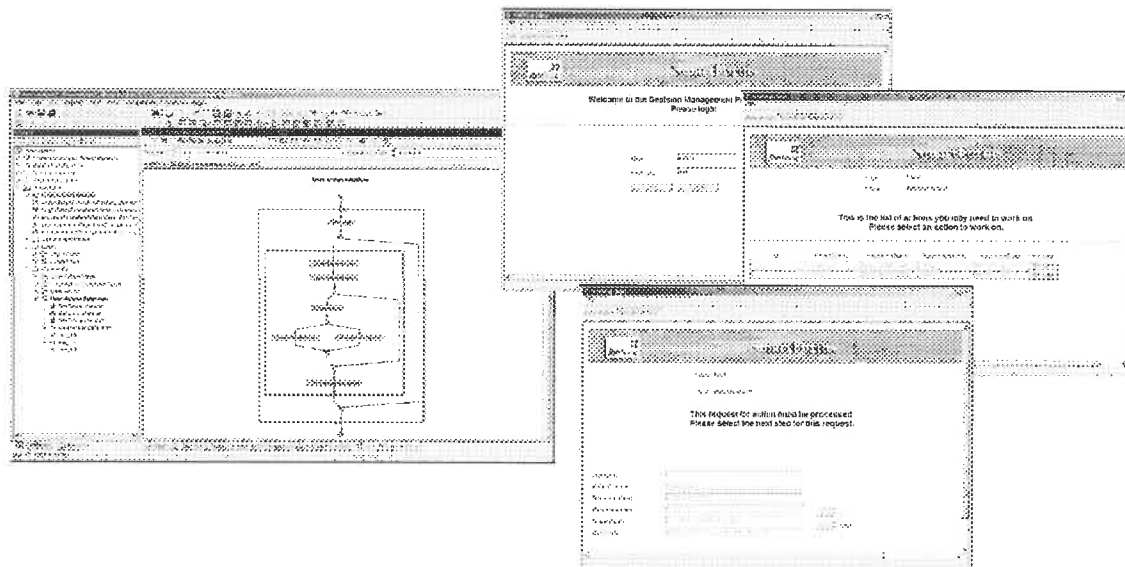


Figure 75—Webflow Configuration of the Workstation

8.5.1.4 FICO Lifecycle Management Summary

FICO's approach and framework for Lifecycle Management processes can be managed in

- FICO's OOTB reference implementation
- Clientworkflow tools

FICO's Lifecycle Manager provides steps for

- Verification
- BrUnit Validation
- Manual Approval
- Release operation
- Publish operation
- Filter
- email notification
- and more...

Customizations are done via known tools

- Blaze Advisor ruleflow / SRL functions
- SmartForms webflows and forms



8.5.2 FICO Lifecycle Management Console Walkthrough

The default implementation defines a simplified "rules lifecycle" workflow that revolves around the overall verification, unit-testing, business-testing and publication approval activities for the two roles of architect and business analyst.

We'll assume that we have a set up that includes the following:

- **DEV:** Development Blaze Advisor repository in which a few rules service are being managed
- **TEST:** Test Blaze Advisor repository which is used to manage rule services that are expected to be tested by the business analysts
- **PROD:** Production Blaze Advisor repository which is used to manage the rule services that are deployed to the production system

These three repositories need to be known and accessible by the default implementation.

The default implementation provides the following:

- 1) a "rules lifecycle" database which keeps track of
 - a) the environment
 - i) which repositories are accessible and how
 - ii) which role they play (development, test, production)
 - iii) which projects in which repository are involved
 - iv) which users are registered with the system and what role they play (architect, business analyst)
 - b) lifecycle processes
 - i) which lifecycle processes have been started on which projects
 - c) action queues
 - i) which actions for which process are to be executed by a given person (architect or business analysts)
 - ii) which actions for which processes are to be executed by an automated configurable service on behalf of architects or business analysts
- 2) a "rules lifecycle" Blaze Advisor service which implements, using Blaze Advisor rule flows and artifacts, the actions on the actual repositories and projects on behalf of the process, as well as the decision on what to the next step in the lifecycle process is
- 3) a "lifecycle console" SmartForms service which implements a basic rudimentary management console for the roles identified above.



FICO Recommendations Whitepaper for Chubb Business Rule COE

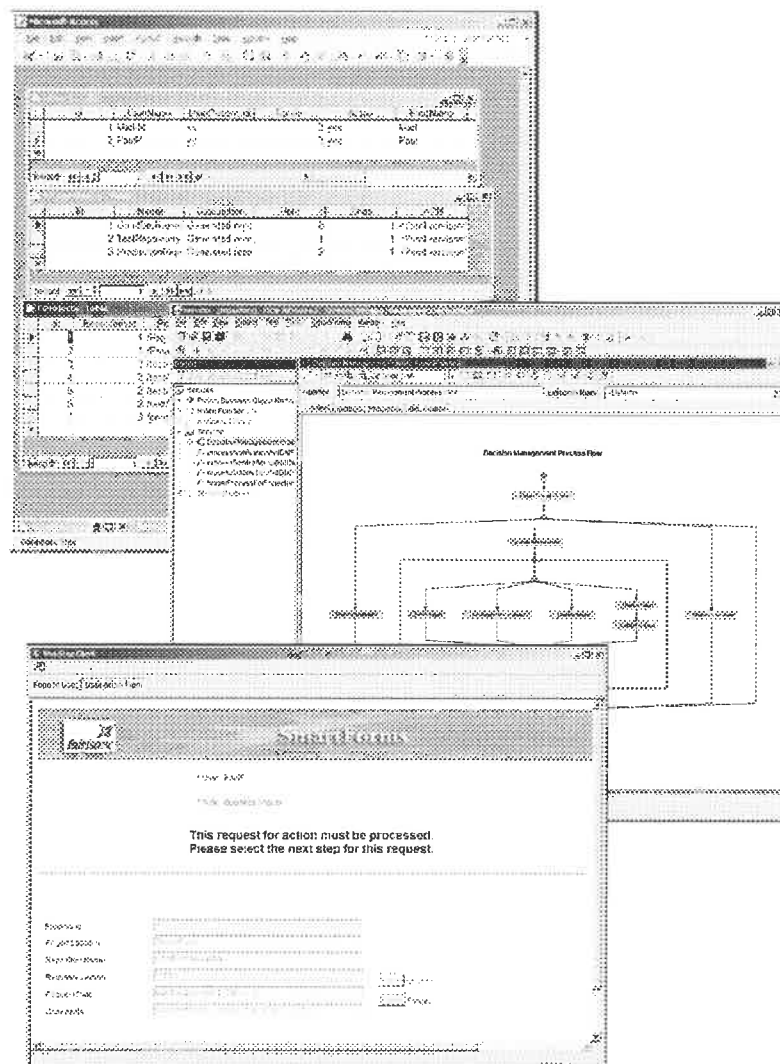


Figure 76—Default Implementation of the FICO Lifecycle Management Workstation (Console)

8.5.2.1 Walkthrough of the FICO Lifecycle Management Workstation (Console)

Imagine a scenario in which two projects (ProjectDev1 and ProjectDev2) need to have their lifecycle managed through this sample workflow.

The application makes the assumption that the requirement to start the lifecycle process on a project first needs to be approved by the business analyst.

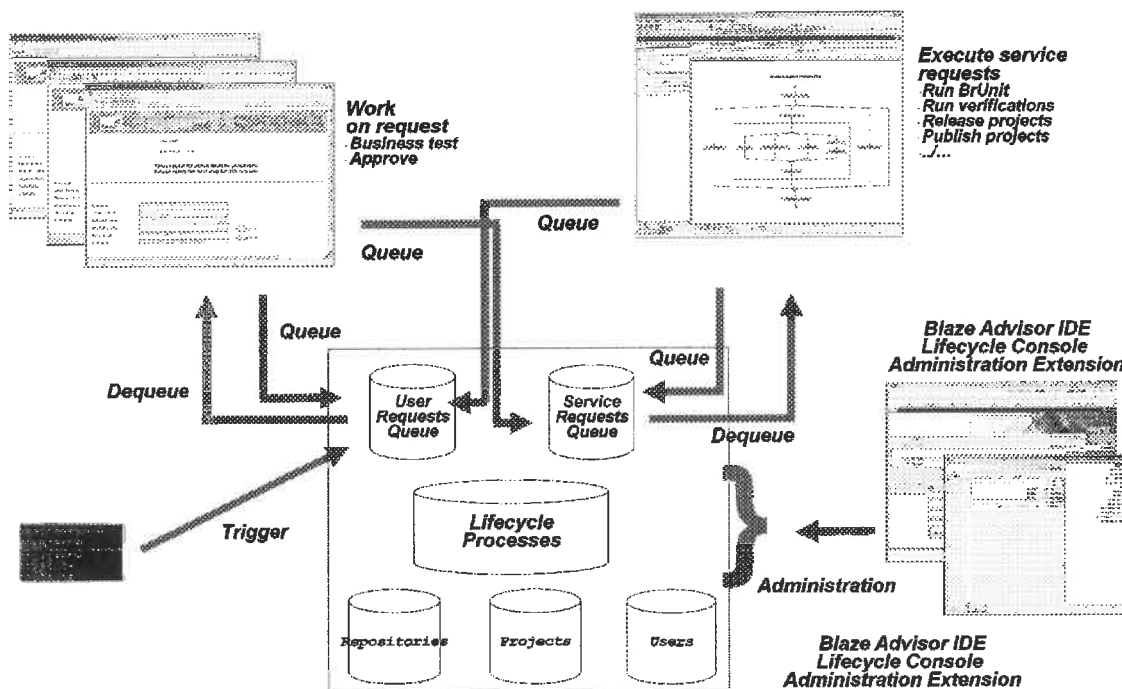


Figure 77—Walkthrough of the FICO Lifecycle Management Workstation (Console)




FICO Recommendations Whitepaper for Chubb Business Rule COE

In the set up that is used for the illustration, PaulP is a business analyst, and we'll log into the SmartForms application that implements the console.

WebClient: Home

Form In Use: Login Form

 **SmartForms**

Welcome to the Decision Management Process console
Please login

User:

Password:

Figure 78—FICO Lifecycle Management Console Login



After his/her login, the business analyst will be presented with a screen that presents the list of all the processes that require the attention of his/her role. Note that all business analysts will get the same screen, and that the selection of a particular process to work on will be guaranteed to be made to only one of them at the time.

WebStep Client

Form in Use: User Action Selection Form

SmartForms

User: PaulP
Role: Business Analyst

**This is the list of actions you may need to work on.
Please select an action to work on.**

ID	ProjectName	RepositoryName	RequestedAction	RequestedDate	Process
1	Project A	Repository A	Requested Action 1	Requested Date 1	Process 1
2	Project B	Repository B	Requested Action 2	Requested Date 2	Process 2

Refresh Cancel

Figure 79—User Action List

You can see that PaulP has two processes to deal with.

The business analyst can click on "Refresh" to get an updated view of the list of requests he/she needs to process, "Cancel" to get out of the application, and "Process" in any of the process rows to select the process to work with.

When "Process" is pressed, the business analyst is brought to another screen in which the details of the process to work with are presented, as well as the list of possible actions with their parameters.

Request Details	
Process Id	1
Project Location	Project A
Repository Name	Project A Repository
Requested Action	Launch
Request Date	2011-03-31 09:15:12 AM
Comments	Project A is now ready to be launched.

Action	Parameters
Launch	
Forget	

Figure 80—Process Action Details

In this particular case, the list of actions is pretty simple but we'll see more complex examples later. If the user presses "Launch", the lifecycle process for this project will be launched.



FICO Recommendations Whitepaper for Chubb Business Rule COE

When the analyst comes back to his processes work items screen, the list is now reduced because the process that was launched was communicated to another role for further processing.

WebStep Client

Form in Use: User Action Selection Form

SmartForms

User: Paul F
Role: Business Analyst

This is the list of actions you may need to work on.
Please select an action to work on.

ID	ProjectName	RepositoryName	RequestedAction	RequestedDate	Process
1	Project A	Repository A	Requested Action	11/11/2010	Process

Refresh Cancel

Figure 61—User Action List

March 31, 2011

FICO Confidential Page 213 of 259

Confidential

FED007129_0213



FICO Recommendations Whitepaper for Chubb Business Rule COE

Let's now switch roles, and consider the case of MarkM who is a rules architect. MarkM may receive a notification warning him that a process was launched and that it requires his attention. This notification may be an email with a link back to his "rules lifecycle" console.

Figure 82—FICO Lifecycle Management Console Login

March 31, 2011

FICO Confidential Page 214 of 259

Confidential

FED007129_0214

The rules architect, who had previously no work to do, is now presented with a worklist that includes the process that was launched.

WebStep Client

Form In Use: User Action Selection Form

fairstep SmartForms

User: Mamta
Role: Architect

**This is the list of actions you may need to work on.
Please select an action to work on.**

Id	ProjectName	RepositoryName	RequestedAction	RequestedDate	Process
1	Project1	C:\Program Files\Microsoft Office\OFFICE11\MSOFTPLT	Edit	12/11/2007	Process

Refresh Cancel

Figure 83—User Action List

You can notice that the rules architect only got to this request one day later, and that the action requested of him is to “verify” the project.

Verifying the project will be done by this role by triggering a Blaze Advisor service that will exercise the product's API, load the project from the information provided in the lifecycle database, and trigger a verification of the rules using the Blaze Advisor services.



FICO Recommendations Whitepaper for Chubb Business Rule COE

The service in question is implemented in the Blaze Advisor repository provided in the default implementation, and processes this request in a very simple way: trigger the verification, record the results and ask for a brUnit unit test to be executed.

WebStep Filter

Form Is Use: User Action Form

Fair Isaac **SmartForms**

User Name: []

Role: Architect

**This request for action must be processed.
Please select the next step for this request.**

Process ID: []

Project Location: []

Repository Name: []

Requested Action: []

Request Date: []

Comments: []

☐ Yes ☐ Cancel

Figure 84—Process Action Details

March 31, 2011

FICO Confidential Page 216 of 259

Confidential

FED007129_0216



FICO Recommendations Whitepaper for Chubb Business Rule COE

Note that when the request to run the unit test (or regression test) is made, the information on how the verification went is provided to inform the decision to proceed with the unit testing.

Figure 60—Process Action Details

In this particular case, the verification failed, because the verifier found a few issues that it reported.

The architect can choose to have those verifications analyzed, or may well decide to proceed with the unit tests in any case. In this case, we'll assume that that is the case.

The architect is then presented with a request to provide an approval to communicate this project to a “test” group by publishing it to the test repository.

When the architect processes that request, he is presented with a screen with information on the results of the unit test run (in this case, one of the unit test cases failed) and how the project should be published to the “test” repository.

The architect can specify where the project should be created (here, the "/test/" folder, which name to give to the published project, and which filter to apply to the publication (here none).

SmartForms

User: [Name]
Role: [Role]

**This request for action must be processed.
Please select the next step for this request.**

Process ID	[Field]	Published Project Location	/test/
Project Location	[Field]	Published Project Name	ProjectDev_1004375726
Request Name	[Field]	Request ID	[Field]
Requested Action	[Field]	Filter Location	[Field]
Request Date	[Field]	Filter	[Field]
Comments	[Field]		

Last Action	[Field]
Results	[Field]
Information	[Field]
Comments	[Field]

Figure 88—Process Action Details



FICO Recommendations Whitepaper for Chubb Business Rule COE

The architect then can come back to his queue of processes to deal with, which now happens to be empty:

WebClient: [Address Bar]

Form In Use: UserAction Selection Form

fair Isaac **SmartForms**

~ User: MarkM
^ Role: Architect

**This is the list of actions you may need to work on.
Please select an action to work on.**

Id	ProjectName	RepositoryName	RequestedAction	RequestedDate	Process

Refresh Cancel

Figure 67—User Action List

March 31, 2011

FICO Confidential Page 219 of 259

Confidential

FED007129_0219



FICO Recommendations Whitepaper for Chubb Business Rule COE

Let's return to the business analyst. The business analyst will now be presented with an additional process to pay attention to: essentially, to test the new project that was pushed into the test environment.

Webstep Client

Form In Use: User Action Selection Form

Fair Isaac **SmartForms**

User: PaulP
Role: Business Analyst

This is the list of actions you may need to work on.
Please select an action to work on.

Id	ProjectName	RepositoryName	RequestedAction	RequestedDate	Process
	SRPProject_1410	FAIRISAC	add	2010-07-14 07:14 PM	Process
	SRPProject_1	FAIRISAC	delete	2010-07-14 07:14 PM	Process

Refresh Cancel

Figure B8—User Action List



FICO Recommendations Whitepaper for Chubb Business Rule COE

If the business analysts proceeds with this process (process 8 in this case), he/she will step through being asked to business-test the project, approve the results of the business tests, which will then result on the project being published to the production environment.

WebStepAction

Form: 164448 User Action Form

SmartForms

User: PaulP
Role: Business Analyst

**This request for action must be processed.
Please select the next step for this request.**

Process ID	164448	Test Result	Passed
Project Location	164448	Test Date	164448
Requestor Name	164448	Test Status	Failed
Requested Action	164448		
Request Date	164448		
Comments	164448		

Figure 53—Process Action Details



FICO Recommendations Whitepaper for Chubb Business Rule COE

Form In Use: 1 User Action Form

SmartForms

* User: FAULP
* Role: Business Analyst

**This request for action must be processed.
Please select the next step for this request.**

*Process ID	1	Published Project Location	prod location
Project Location	1: PROD/PROD/PROD/PROD	Published Project Name	ProjwData_1661075793_1012111238
Repository Name	PROD/PROD	Repository ID	1
Requested Action	PROD/PROD	Filter Location	Proj1FilterTestOut
Request Date	10/12/2011 10:12:38	<input type="checkbox"/> Update	
Comments	<input type="checkbox"/> Reject		

Figure 90—Process Action Details

Note that in this case, the business analyst has chosen to apply the Proj1/FilterTestOut filter to the publication. This filter defined in the “test” project (and before that in the development project) filters out the entities that have the “Status” management property set to “Test”.



FICO Recommendations Whitepaper for Chubb Business Rule COE

The business analyst is then back to his/her process list.

SmartForms

User: Business Analyst

Role: Business Analyst

This is the list of actions you may need to work on.
Please select an action to work on.

ID	ProjectName	Requestor Name	Requested Action	Requested Date	Process
1	Project A	John Doe	Review	2011-03-31	Process A

Refresh Cancel

Figure 91—User Action List

The end result is a project that includes the released project in its "Production" folder.

8.5.3 FICO Lifecycle Management Lifecycle Database

The lifecycle database is a very simple SQL database that can be easily implemented in any database accessible through JDBC.



FICO Recommendations Whitepaper for Chubb Business Rule COE

You will implement a database that looks as follows:

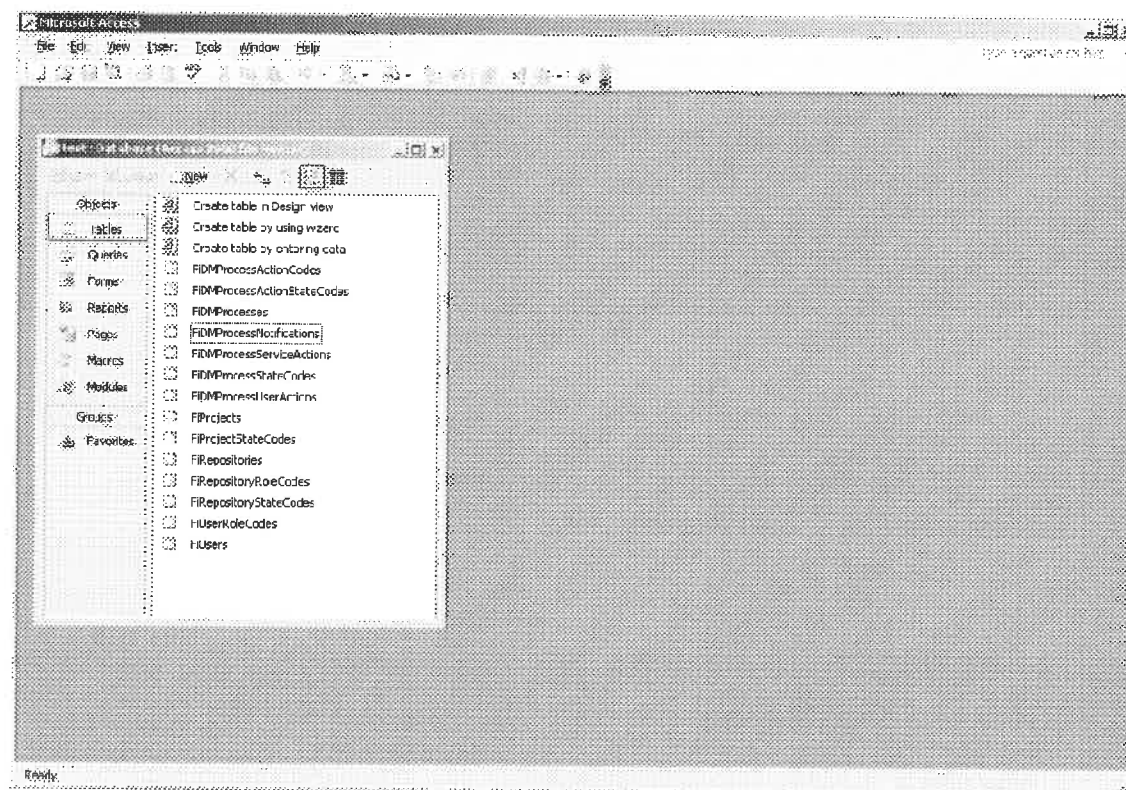


Figure 92—FICO Lifecycle Management Lifecycle Database

Note that the tables in the database have structure that you can take a look at, but not modify. The "...Codes" tables are pre-populated with information that is used by the default implementation.

March 31, 2011

FICO Confidential Page 224 of 259

Confidential

FED007129_0224



FICO Recommendations Whitepaper for Chubb Business Rule COE

Of particular note are the FiDMProcessUserActions and FiDMProcessServiceActions tables which contain the process queues for users and for the "rules lifecycle" Blaze Advisor service requests.

Id	Processed	Poid	RequestedAction	RequestDate	Comments	Role	Processed
1	1			1 Tue Mar 27 19:5	Launched on Tue Mar 27 19:5	2	
6	1			7 Tue Mar 27 19:4	Requested by service after	2	
7	1			2 Tue Mar 27 19:4	Requested by service after	2	
8	22			3 Tue Mar 27 19:5	Requested by service after	3	
9	22			4 Tue Mar 27 19:5	Request on Tue Mar 27 19:5	3	
1	1	1		0 Mon Mar 26 21:	Triggered on Mon Mar 26 21:	3	
4	4	1		0 Mon Mar 26 21:	Triggered on Mon Mar 26 21:	3	0

Figure 93—Process User Actions Table

March 31, 2011

FICO Confidential Page 225 of 259

Confidential

FED007129_0225



FICO Recommendations Whitepaper for Chubb Business Rule COE

You need to at the very least populate the FiUsers, FiRepositories and FiProjects tables. These tables contain the information on the users and their roles, the repositories and their roles, and the projects.

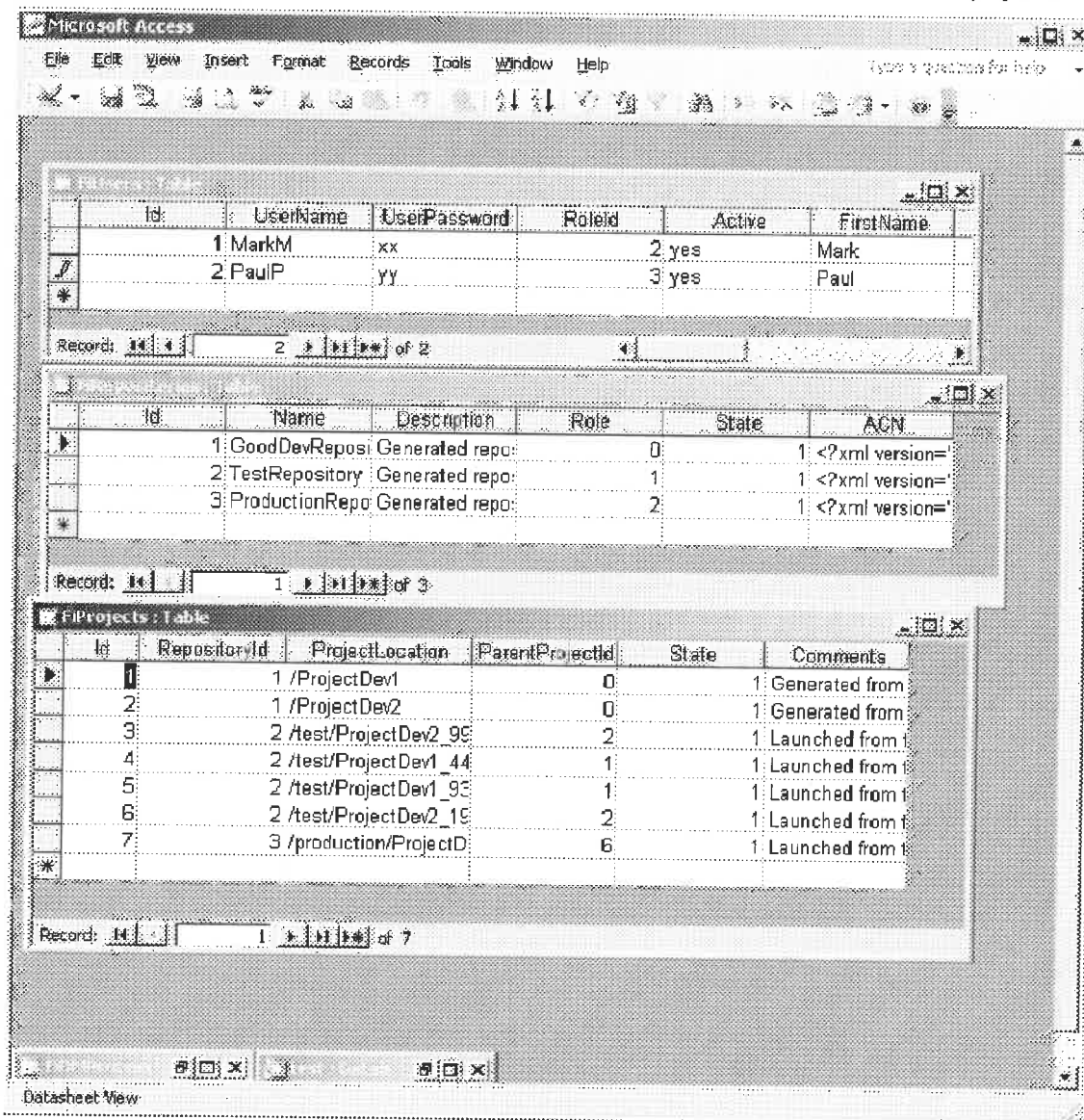


Figure 84—Framework Configuration Tables

The database can be populated through an extension to the Blaze Advisor IDE that you can use to manage it once created.



FICO Recommendations Whitepaper for Chubb Business Rule COE

The extension is installed by using the NdLifecycleExtension.xml extension file you will find in your installation. Once you install it and re-start the Blaze Advisor IDE, you will find a new command in the "Repository Administration" menu.

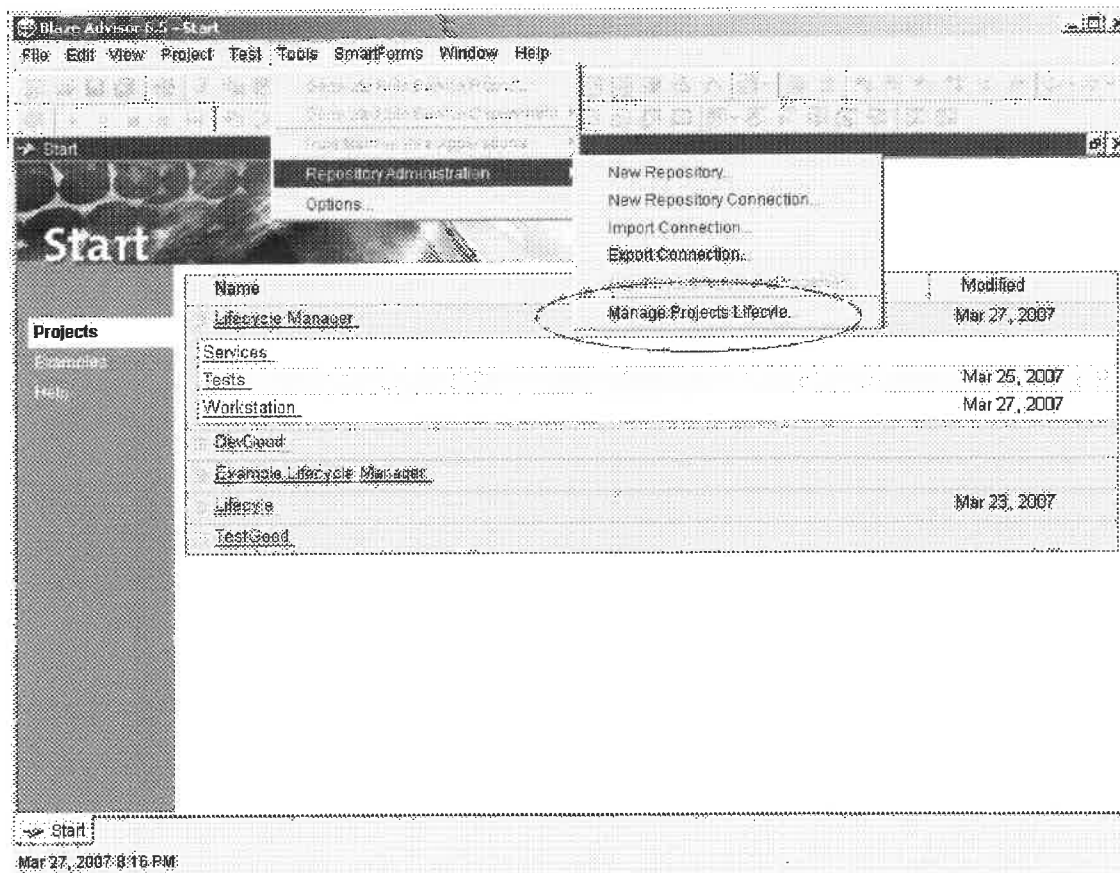


Figure 95—Extensions to the Blaze Advisor IDE



FICO Recommendations Whitepaper for Chubb Business Rule COE

You will be asked to login to the database you created:

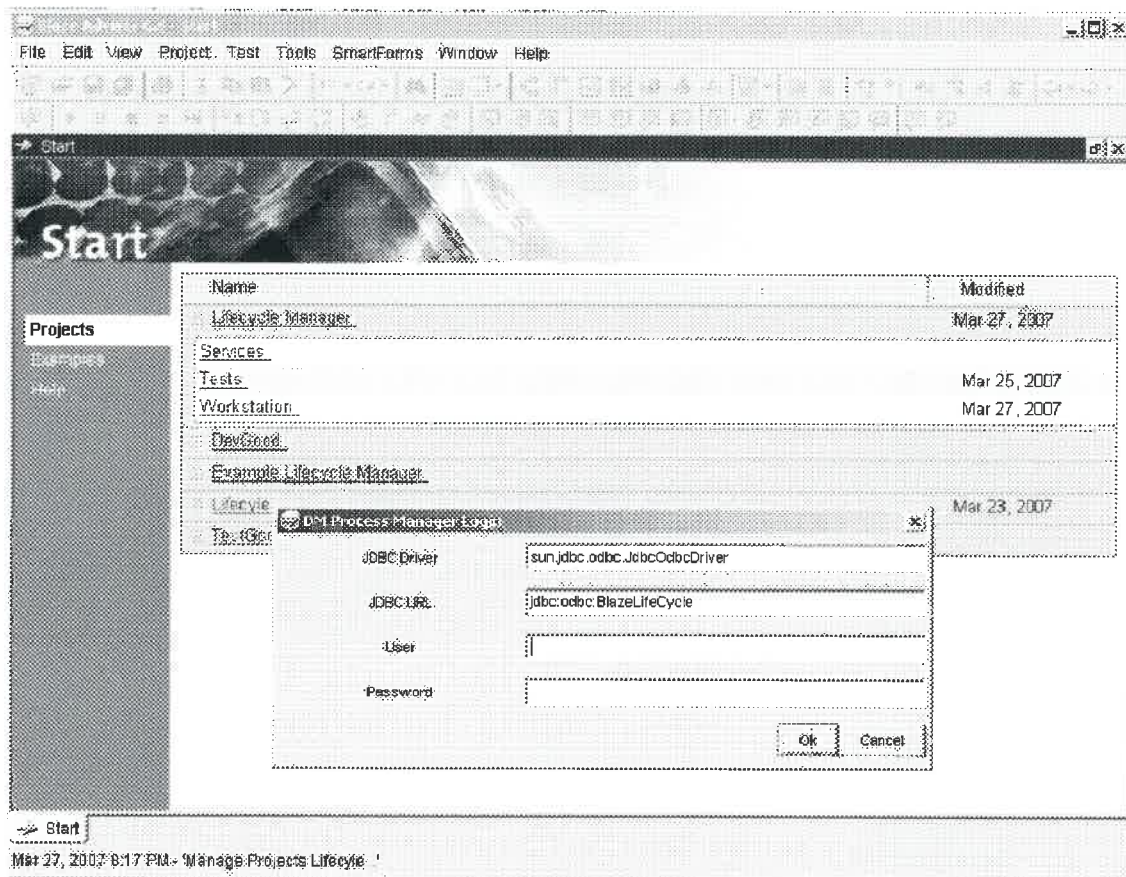


Figure 56—Login to FICO Lifecycle Management Lifecycle Database



FICO Recommendations Whitepaper for Chubb Business Rule COE

And you will be presented with an options dialog which lets you browse through and edit parts of the lifecycle database.

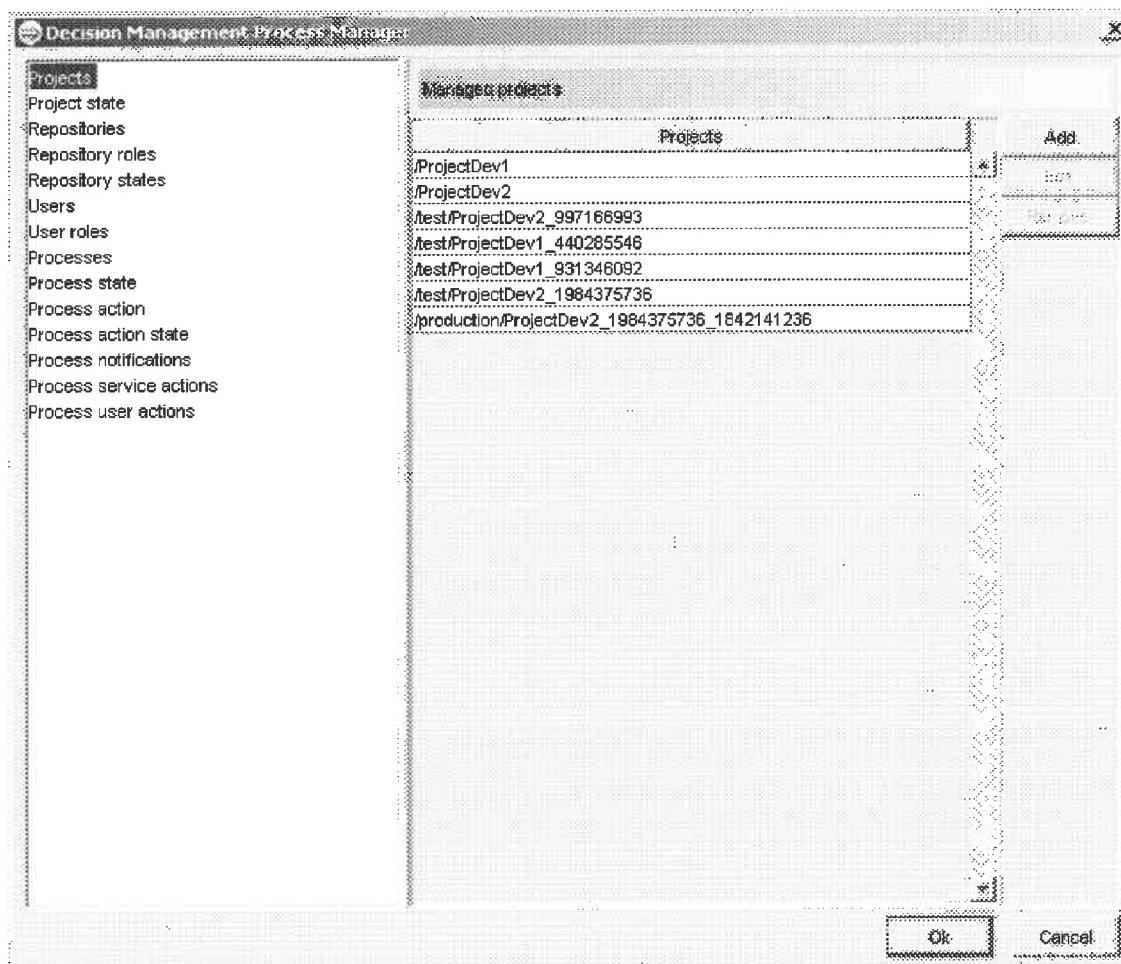


Figure 97—Decision Management Process Manager



FICO Recommendations Whitepaper for Chubb Business Rule COE

Adding a user is just a question of going to the “Users” option, press the add button and fill the corresponding information. Once “Ok” is pressed in the dialog, the information is written back to the database.

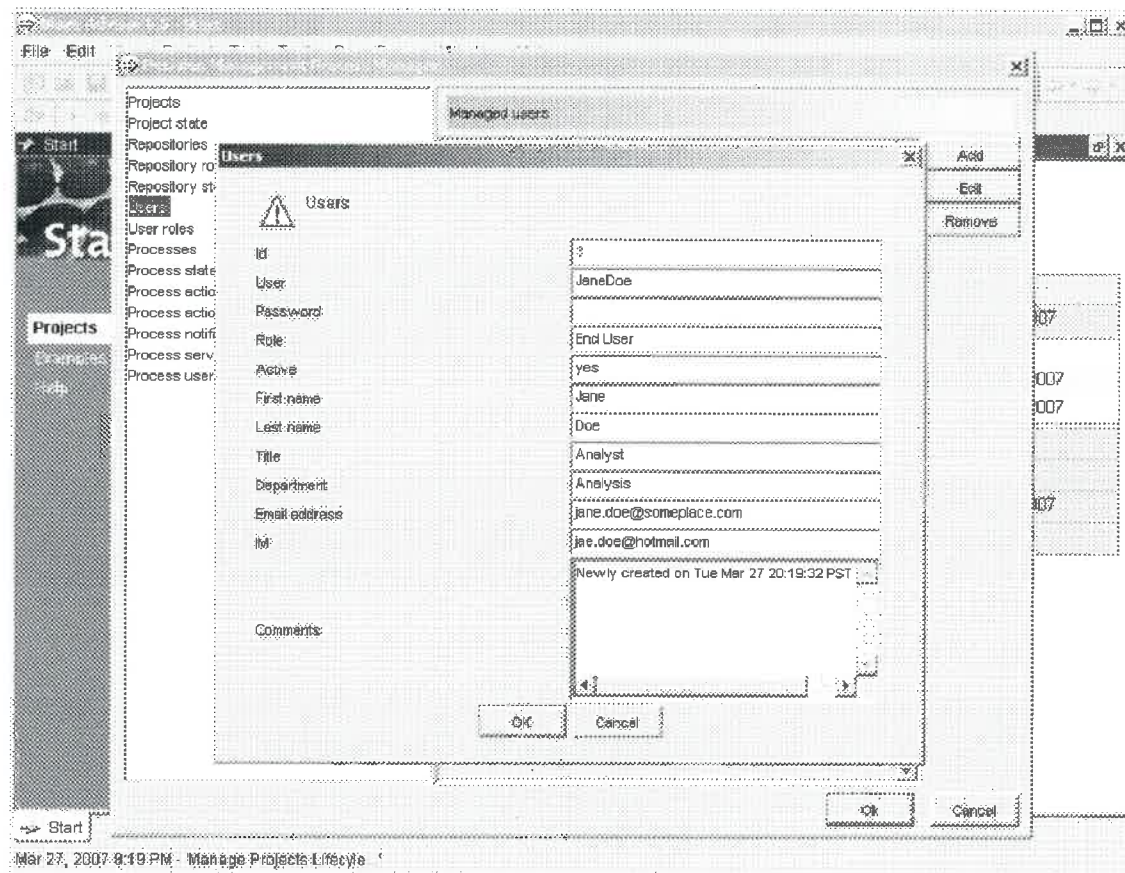


Figure 90—Adding a User



The same is true with repositories. In addition, if you are connected to a repository, creating a repository through the dialog will have the information for the currently loaded repository pre-loaded.

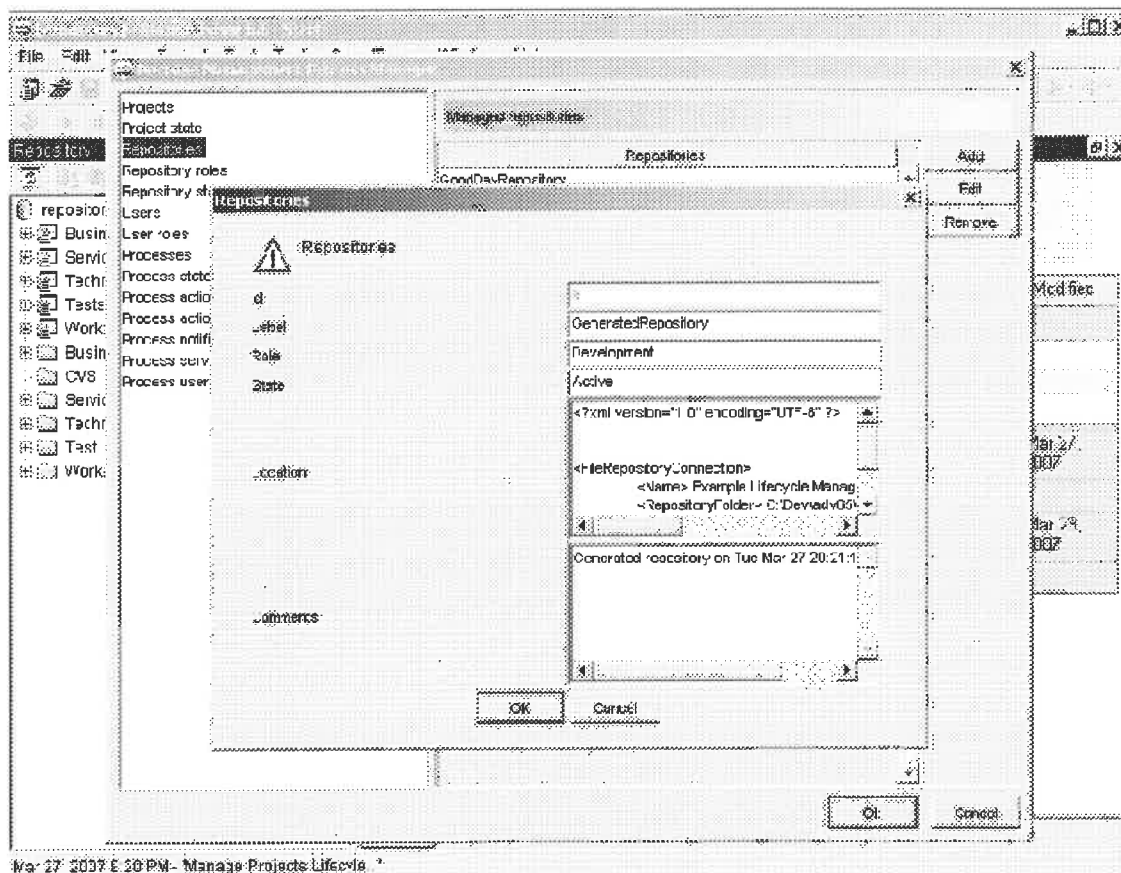


Figure 88—Adding a Repository



FICO Recommendations Whitepaper for Chubb Business Rule COE

And, the same is true for projects.

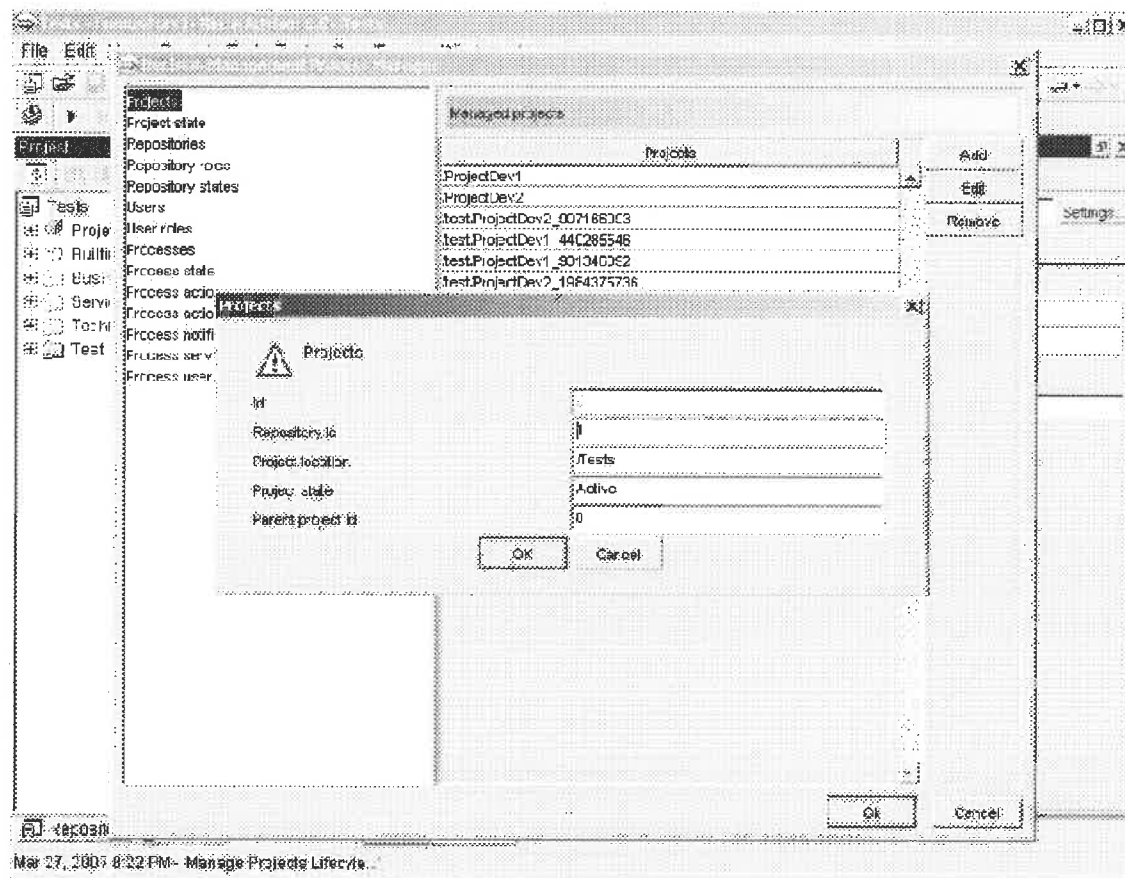


Figure 100—including a Project

This extension can be used to get started with the initial population of the “Lifecycle database” with the initial information.



8.5.4 Lifecycle Management Rule Service

The “Lifecycle Management” rule service is implemented through a traditional Blaze Advisor rule service, which takes advantage of the high level extensions object model that leverages the lifecycle APIs provided by the 6.5 release of the product.

The “repository” repository in this default implementation follows the standard structure, organized in a “Technical Library” maintained by the rules architects, a “Business Library” which contains all the decisions maintained by the business users, and a “Services” library in which the orchestration of the logic and the service entry points are defined.

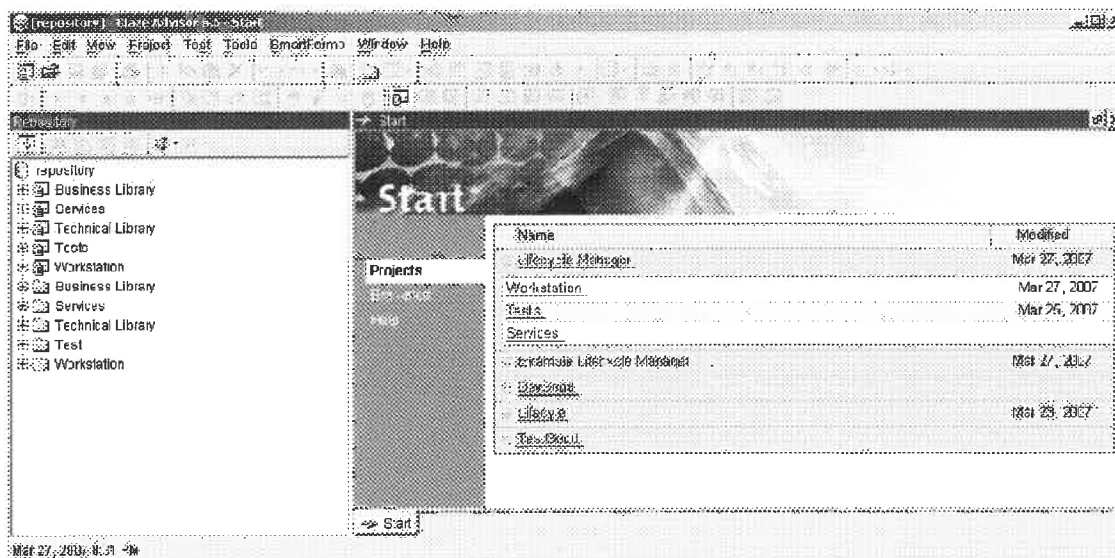


Figure 101—Services and Workstation Projects under FICO Lifecycle Manager

8.5.4.1 Business Object Model

The business object model is implemented in the `com.blazesoft.devtools.extensions.lcycle.data` package (object representation of the information in the “Lifecycle database”) and the `com.blazesoft.devtools.extensions.lcycle.management` package (implementation of the various process, project and repository lifecycle management high level APIs).



FICO Recommendations Whitepaper for Chubb Business Rule COE

You can find it through the “Technical Library” project.

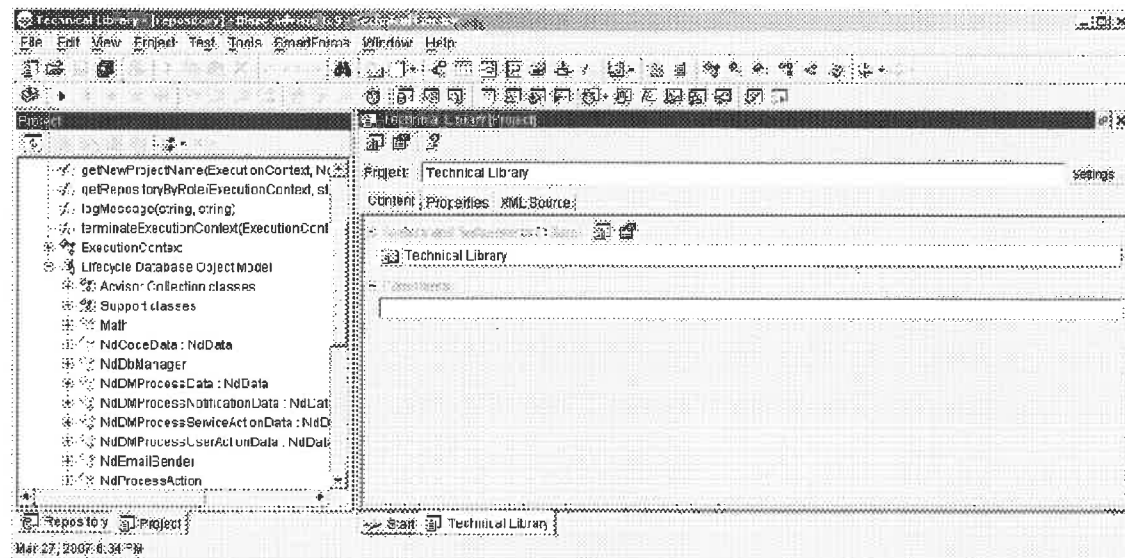


Figure 102---Business Object Model



8.5.4.2 Overall Decision Flow

The overall decision flow for how to process the requests to the service and how to decide what to do next is implemented through the rule flow accessible in the “Services” project.

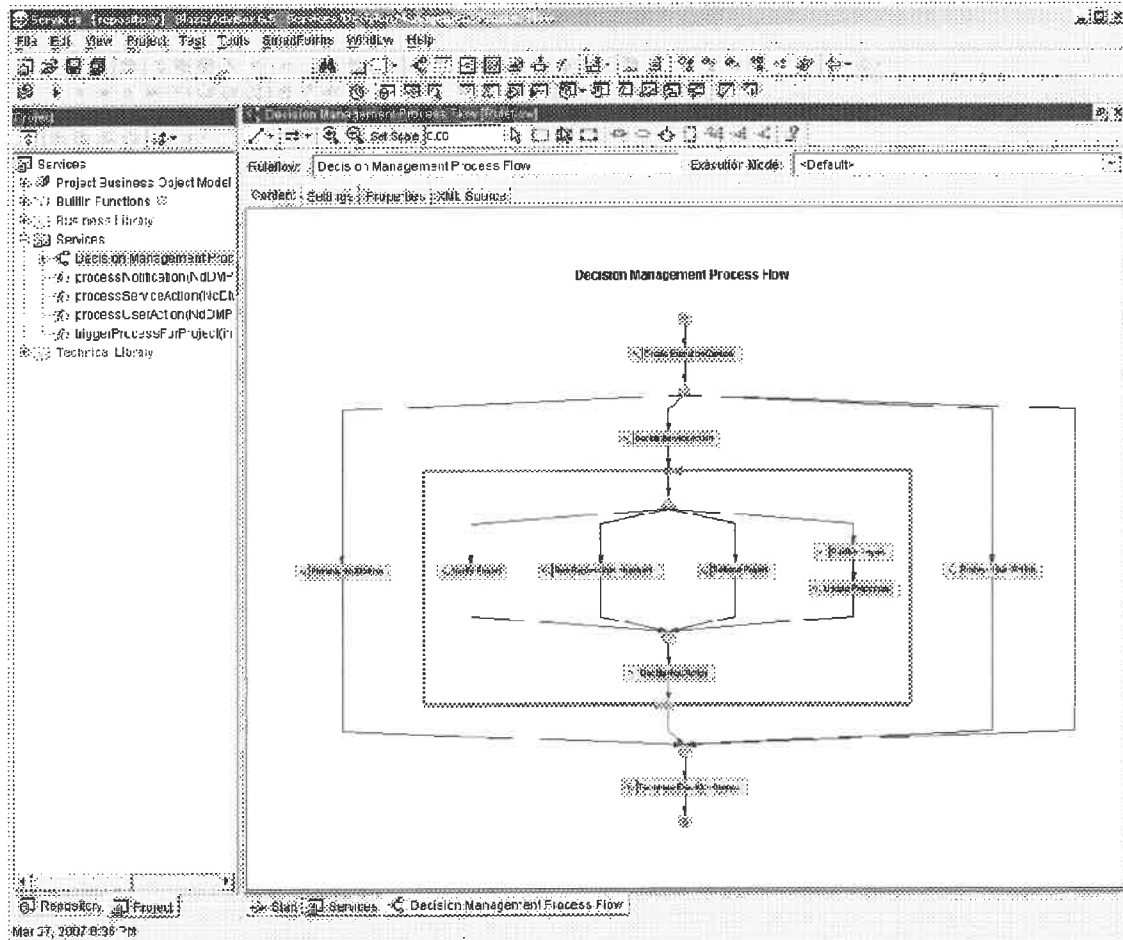


Figure 103—Process Request Workflow

The flow simply follows these steps:

- 1) Introspect the request and decide on the request
 - a) If the request is simply a notification, do nothing
 - b) If the request is a call for user action, simulate the response (this section should be removed in a real deployment)
 - c) If the request is a call for the service to execute an action, decide on how to execute it and let the flow route the work to the proper activity
- 2) In the case work was routed to an activity, have it do its work, and then potentially send notifications to keep the other entities up to date



- 3) Decide what to do next, and translate that into posting requests for action to users (identified through role), service (effectively calling itself or another instance of itself), or nothing.

Because the interactions are with databases, the processing is managed in the context of a connection that is obtained and closed at the beginning and end of the flow respectively (with exception management in the flow's abort section).

It should be noted that this flow is modifiable by the users. Care should be applied to make sure the processes are completely managed.

8.5.4.3 Business Library

The implementation of each one of the activities is provided in the "Business Library". These functions invoke the lifecycle API to do their work.

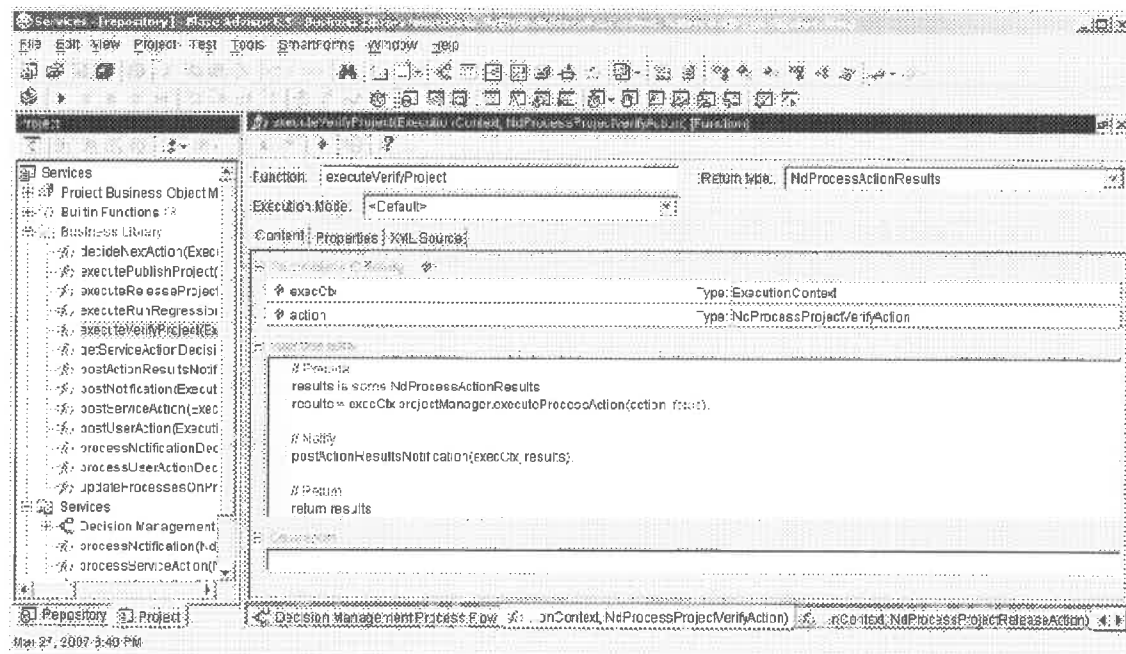


Figure 104—Business Library Functions



FICO Recommendations Whitepaper for Chubb Business Rule COE

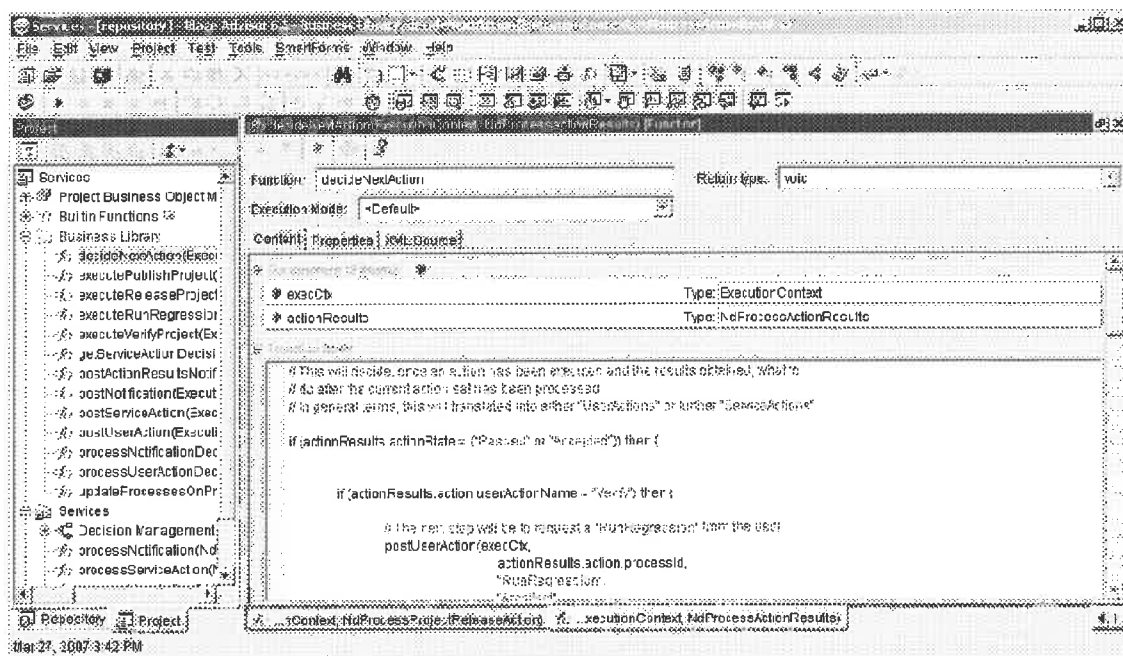


Figure 106—Business Library Functions

In general terms, these are very simple functions or rulesets, and offer ample possibilities to customize and templatize.

8.5.4.4 Deployment

The service is deployed by creating a standard Blaze Advisor rule service that exposes one or more of the functions defined in the “Services” project as entry points.

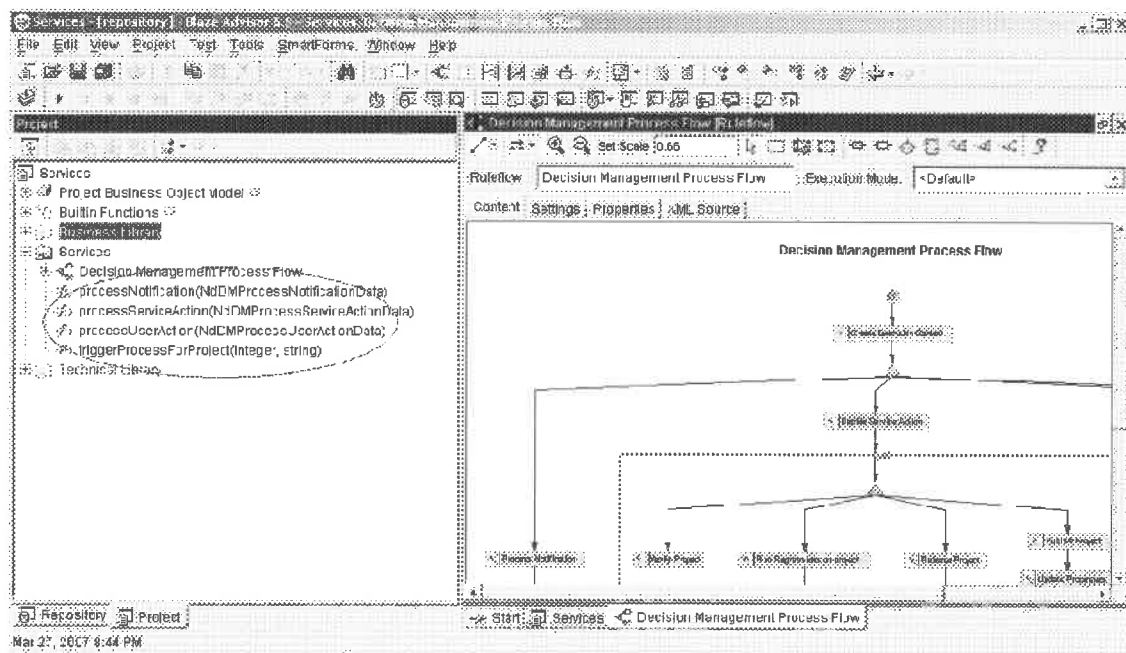


Figure 106—Services Project Deployment

The corresponding entry services will expect as an input an instance of a `com.blazesoft.devtools.extensions.lcycle.data` object. The way that object is constructed is not important to the implementation of the service, but it is expected that each instance corresponds to a request for action extracted from the database.

In general, you are responsible for the deployment of this service and its integration with an environment that extracts information from the database, construct the appropriate data objects and feeds them to the service.

For convenience the default implementation includes in the “services/” folder a Blaze Advisor server deployment descriptor. That descriptor refers to an example of a service deployed as a simple Java class that polls that database a pre-defined rate, extracts data objects and feeds them to the service. You can use it as is by launching it from the “services/” subfolder of the default implementation, with all Blaze Advisor “jars”, including the development ones, in the CLASSPATH:

```
java com.blazesoft.devtools.extensions.lcycle.services.NdProcessLifecycleService
```

or you can subclass that class, create your own main function, or create your own implementation. In the latter cases, remember to either generate your own



FICO Recommendations Whitepaper for Chubb Business Rule COE

The code for the service follows.

```
//
// Blaze Advisor Server Deployment.
// Generated by the Blaze Advisor Quick Deployer
//
// Copyright (1997-2006), Fair Isaac Corporation. All Rights Reserved.
// All Rights Reserved.
//

package com.blazesoft.devtools.extensions.lcycle.services;

import com.blazesoft.server.base.NdServiceSessionException;
import com.blazesoft.server.base.NdServiceException;
import com.blazesoft.server.base.NdServerException;
import com.blazesoft.util.NdWrappedExceptionSupport;
import com.blazesoft.util.NdWrappedRuntimeException;
import java.io.*;
import java.util.*;
import com.blazesoft.devtools.extensions.lcycle.data.NdDbMonitor;
import com.blazesoft.devtools.extensions.lcycle.data.NdDbMonitorProcessorBase;
import com.blazesoft.devtools.extensions.lcycle.data.NdDbManager;
import com.blazesoft.devtools.extensions.lcycle.data.NdDMPProcessUserActionData;
import com.blazesoft.devtools.extensions.lcycle.data.NdDMPProcessServiceActionData;
import com.blazesoft.devtools.extensions.lcycle.data.NdDMPProcessNotificationData;

/**
 * This class runs an example of the server deployment generated by the Advisor
 * QuickDeployer.<p>
 *
 * This class is provided as a self-contained example of:
 * <ul>
 * <li>How an Advisor Server Java deployment can be constructed.
 * <li>How clients can invoke the server's entry points.
 * </ul>
 */

public class NdProcessLifecycleService extends NdDbMonitorServiceBase
{
    // Private data
    // -----

    // Server configuration file name
    private final static String _SERVER_CONFIG =
        "./ProcessLifecycleServer.server";
    private NdProcessLifecycleServer _server;
    private boolean _monitorQueues;
    private String[] _queueRoles;

    /**
     * Constructor
     */
    public NdProcessLifecycleService(NdDbManager dbManager, boolean monitorQueues,
        String[] queueRoles, long timeOut)
    {
        super(dbManager, timeOut);

        try {
            // Create the server

```

March 31, 2011

FICO Confidential Page 239 of 259

Confidential

FED007129_0239



FICO Recommendations Whitepaper for Chubb Business Rule COE

```

        String serverConfig = _SERVER_CONFIG;
        _server =
            (NdProcessLifecycleServer)NdProcessLifecycleServer.createServer(s
            erverConfig);
        _monitorQueues = monitorQueues;
        _queueRoles = queueRoles;
    }
    catch (Exception ex) {
        throw new NdWrappedRuntimeException(ex);
    }
}

protected void extractAndProcess() throws Exception
{
    // Try to get a service action request first
    NdDMProcessServiceActionData serviceActionData =
    getDbManager().getDMProcessServiceActionsDb().retrieveNonProcessedProce
    ssServiceAction();
    if (serviceActionData != null) {
        // Process
        _server.invokeProcessServiceAction(serviceActionData);
    }
    // Try to see whether there are user actions that this should be
    listening to
    if (_monitorQueues) {
        for (int i = 0; i < _queueRoles.length; i++) {
            NdDMProcessUserActionData userActionData =
            getDbManager().getDMProcessUserActionsDb().retrieveNonProc
            essedProcessUserAction(_queueRoles[i]);
            if (userActionData != null) {
                // Process
                _server.invokeProcessUserAction(userActionData);
            }
        }
    }

    // Finally look at notifications
    NdDMProcessNotificationData notificationData =
    getDbManager().getDMProcessNotificationsDb().retrieveNonProcessedProces
    sNotification();
    if (notificationData != null) {
        // Process
        _server.invokeProcessNotification(notificationData);
    }
}

protected void processorInitialized() throws Exception
{
}

protected void processorEnded() throws Exception
{
    try {
        // Shut down the Server server instance
        _server.shutdown();
    }
    catch (Exception ex) {
        throw new NdWrappedRuntimeException(ex);
    }
    getDbManager().close();
}

```

March 31, 2011

FICO Confidential Page 240 of 259



```

protected void processException(Exception e)
{
    super.processException(e);
    // Report to console...
    System.err.println(e.toString());
    e.printStackTrace();
}

/**
 * Main method. triggers the service
 */
public static void main(String[] args)
{
    /** @localization off */
    NdDbManager dbManager =
        NdDbManager.createNewDbManager("sun.jdbc.odbc.JdbcOdbcDriver",
        "jdbc:odbc:BlazeLifeCycle", "", "");
    String[] roles = new String[] { "Administrator", "Sys Admin",
        "Architect", "Business Analyst", "End User", "Product Manager" };
    NdProcessLifecycleService service = new
        NdProcessLifecycleService(dbManager, false, roles, 500L);

    /** @localization on */
    service.start();
}
}

```

Again, this is just an example.

Any validly licensed Blaze Advisor deployment that is hooked to a system that reads the new requests for action from the database (those with Processed set at 0), extracts them into the corresponding data objects, and invokes the corresponding service (as above) will be appropriate.

8.5.4.5 Lifecycle Console

The Lifecycle console requires the SmartForms for Blaze Advisor 6.5 add-on to be installed and enabled.



FICO Recommendations Whitepaper for Chubb Business Rule COE

Assuming it is enabled, you will find the implementation of the console in the “Workstation” project in this repository.

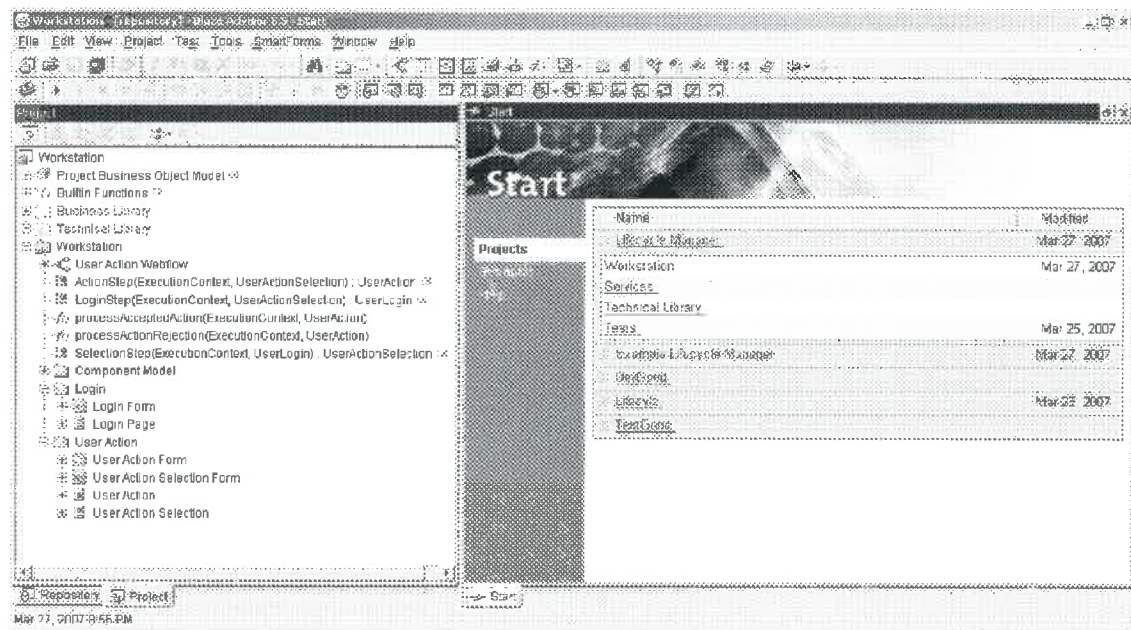


Figure 107—Services and Workstation Projects under FICO Lifecycle Manager



The Login form, and the User Action Selection and User Action forms are easily found in the corresponding folders. Note that these forms are tied to an XML-based component model that can be found in the class provider in the Component Model folder. The model in question is designed to convey to the forms the information they need to display, and convey back to the WebFlow the information the user enters or selects.

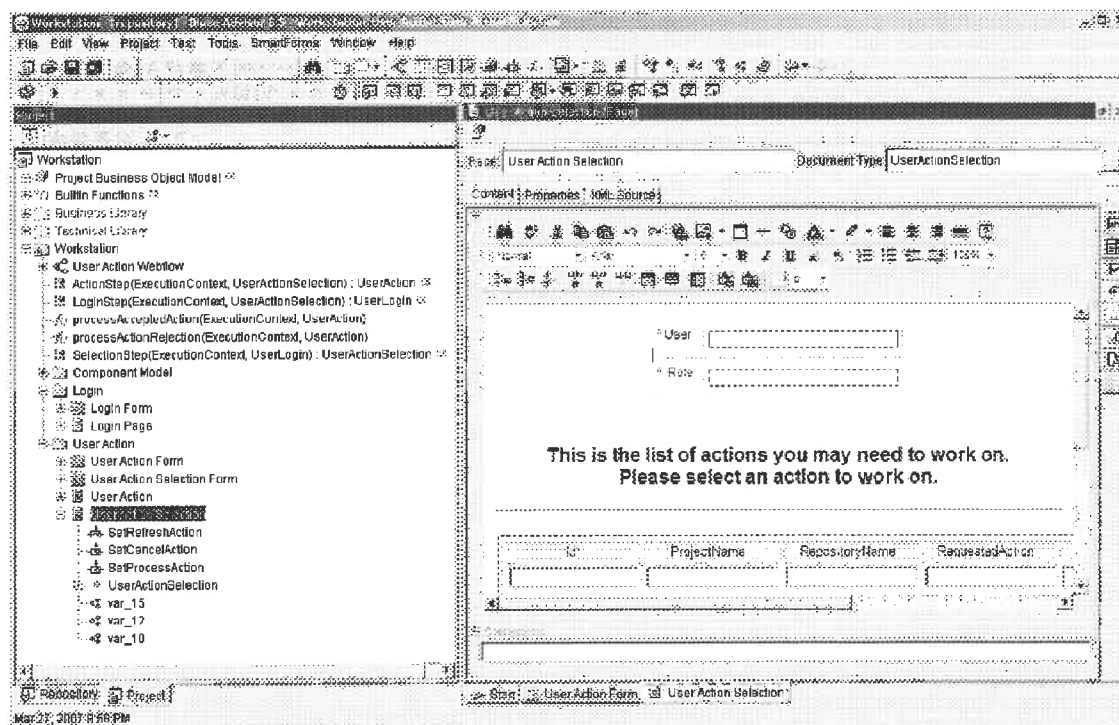


Figure 103—Creating the Workstation (Console) Login Form

Note that the SmartForms implementation takes advantage of the “forms rules” capabilities of the tool to simplify its implementation.

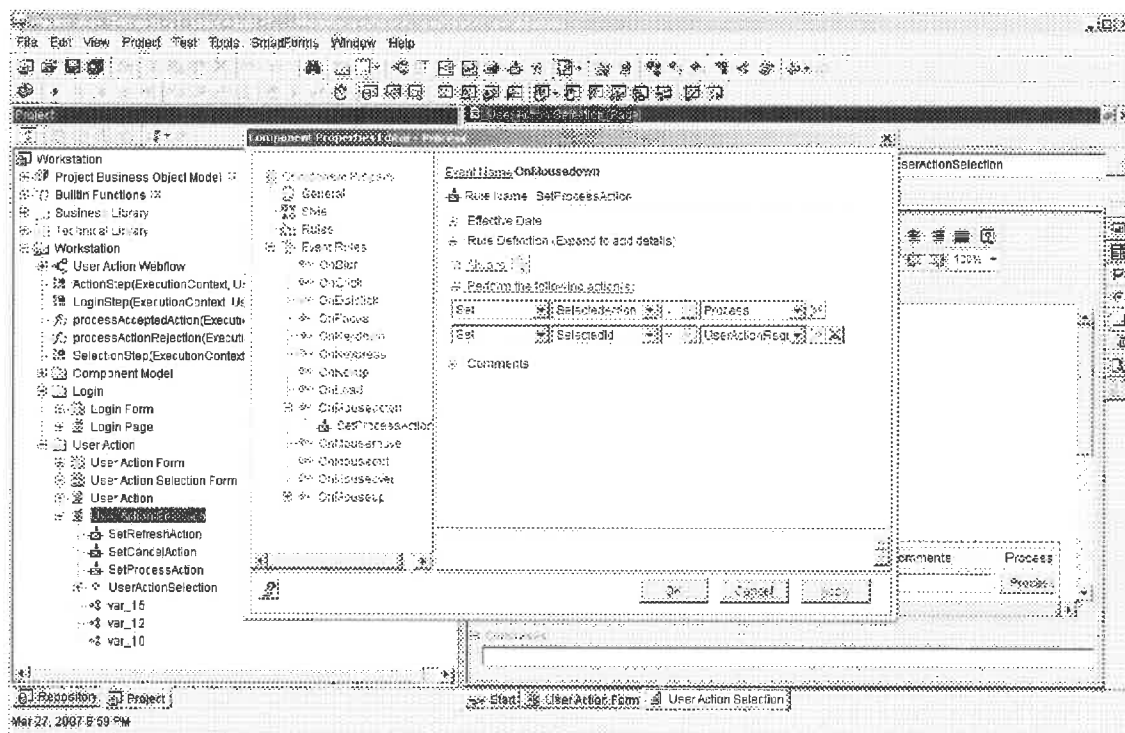


Figure 103—SmartForms Process Event Configuration



The overall WebFlow orchestrates the forms, filling the information based on the previous form as well as the intermediate steps, and applies back end logic to decide what the next steps should be.

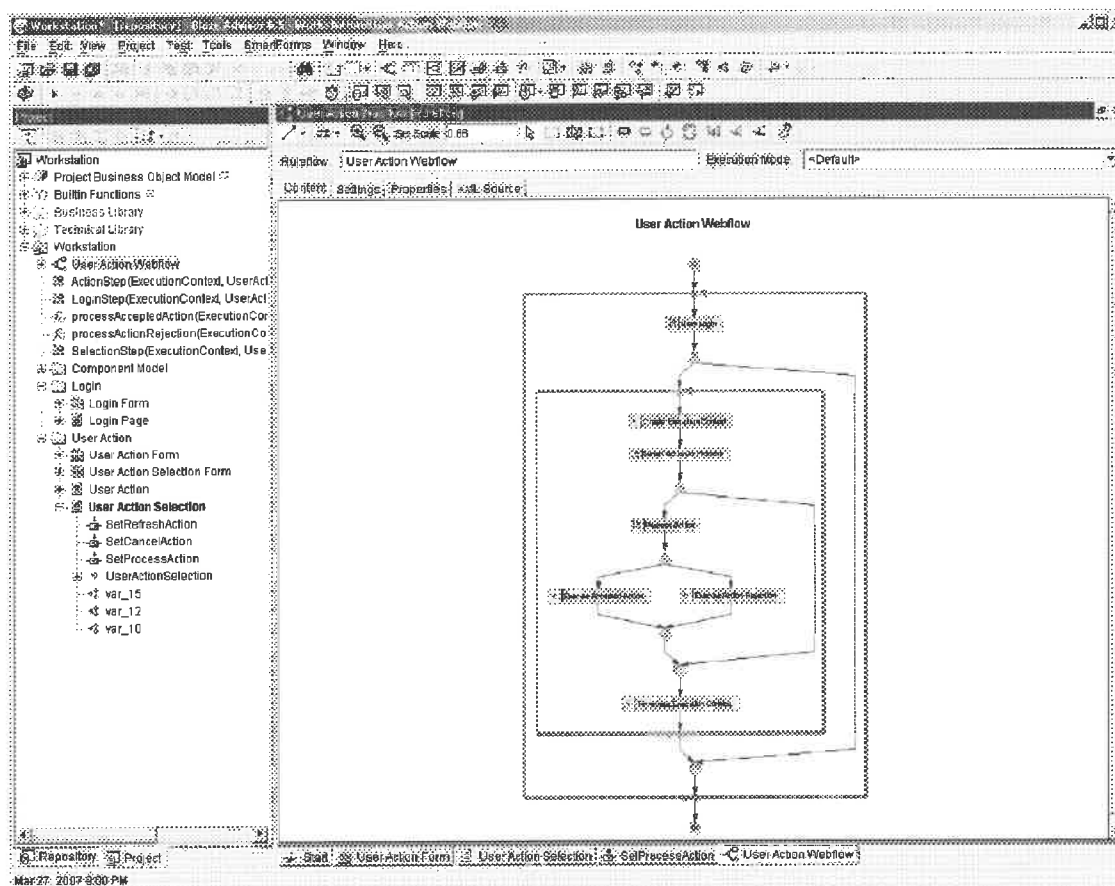


Figure 110—Process User Action Webflow

In essence, the WebFlow is the user-centric equivalent to the “rules lifecycle” Blaze Advisor service covered earlier.

The WebFlow can be deployed like any standard SmartForms for Blaze Advisor WebFlow. The Blaze Advisor “jars”, including the development ones, need to be accessible to the deployment.

8.5.5 Best Practices and Considerations

FICO's Lifecycle Management Service and Console is an innovative feature uniquely available today with Blaze Advisor. Allows BRMS solution to extend rules management concept well beyond authoring.

Lifecycle Management is based on Blaze Advisor's published framework that can be customized to fit the enterprise needs. It is very powerful when combined with the other lifecycle management features

- Verification
- Validation



8.5.5.1 Best Practices

The following are some pertinent best practices followed by FICO's Professional Services in their implementations of the Lifecycle Management framework.

1. Analyze early on the governance imperatives and the associated processes
 - Who owns what
 - Who may modify the same entities
 - At what stage in the process
 - How are changes rolled into Production
 - Who needs to physically approve those rule changes
 - What needs to be documented for audit purposes
 - Etc.
2. We recommend structuring the virtual enterprise repository into multiple areas
 - Development: Technical and/or Business authoring
 - Testing: System and/or performance testing
 - Production
 - And any other that make sense for the business
3. Individual environments do not need to use the same storage mechanism
 - They could be file-based, database... or simply an ADB file
 - They may not share the same versioning / AAC mechanisms
 - They could be (and typically are) on different machines
4. Consider the many options for change management depending on the enterprise specific needs
 - Should all flavors of the same rule over time be active at any point in time?
 - What is the level of granularity? Sub-project? Ruleset? Rule?
 - What kind of processes need to be deployed?
 - Any patches in Production that need to be applied to Development?
 - What are the company policies in terms of tracking for retired rules?
 - Etc.

8.5.5.2 Design Considerations

While the approach outlined above is powerful and effective, it does not represent an "out of the box" (OOTB) approach to lifecycle management using Blaze Advisor. There are two key considerations that must be measured prior to taking this approach:

1. SmartForms are not integrated into the Blaze Advisor RMA and therefore do not have ready access to the OOTB version management features provided with the Blaze RMA. These include things like the visual comparison tool that allow the RMA user to check how repository entities may have changed before promoting them.
2. FICO's SmartForms product is licensed separately from Blaze Advisor. An additional license agreement will be required to use the SmartForms approach.



3. SmartForms in itself is NOT a lifecycle management or workflow management tool. It is a tool developed by FICO for the purpose of embedding BRMS intelligence within web forms. The framework described above uses certain functionalities from SmartForms. However, it can also be implemented without SmartForms if the project is already employing alternate technologies.
4. FICO PS does not require the purchase of a SmartForms license to implement this solution framework.

8.6 The “Maker – Checker” Approach

FICO PS has developed an approach to managing some of the unfavorable side-effects from extensive use of management properties in large BRMS projects that are very volatile or undergo a high rate of maintenance. This section provides details on this approach.

8.6.1 Overview

There are two important considerations in using management properties to manage and control business rule lifecycle in large BRMS projects.

1. Repository size due to unnecessary version management overhead
2. Lack of extensive features and flexibility in the editing metaphors requires architects to keep the design simple.

8.6.1.1 Repository Size—Unnecessary Version Management

The first consideration results from side effects of the version management mechanisms within the Blaze Advisor repository. If you are planning to use the Blaze Versioning System in conjunction with Management Properties, you need to consider the following:

- The values for management properties are stored along with the instances of their corresponding repository entities.
- Every time a management property is changed to signify approval for release to the next SDLC stage, the entity must be checked-out, modified, and then checked back in.
- From the perspective of BVS, any change to the value of a management property is a change to the instance of the repository entity. Therefore, a new version of the repository entity will be created, even if no other changes were made to the entity.

The above can occur many times as repository entities change status as they migrate through the various SDLC stages (e.g., DEV → TEST → QA → DEPLOY). In this example, there will be a minimum of four versions of every repository entity in the repository. In large volatile implementations, the size of repositories using this approach can quickly grow out of control.

8.6.1.2 Editor Flexibility—Keep the Design Simple

The second consideration in using management properties is the editor metaphor for management properties in the RMA in the current version of Blaze Advisor (version 6.10).

- The management property editor cannot reference values from other value holders within the instance of the repository entity (i.e., the rule name).
- Access to management property values cannot be controlled programmatically.

These last two considerations are indeed minor and rarely result in barriers to implementing functional requirements. They are only mentioned because some architects attempt to get overly creative in their designs. Keep the design simple and the editor functionality will be sufficient.



8.6.2 “Maker—Checker” Approach

FICO Professional Services, in collaboration with several of our clients, has devised an approach called “Maker—Checker” that overcomes some of the management property limitations and substantially improves the scalability of lifecycle management using management properties.

The basic principle behind maker-checker is one person (or role) checks out a file and makes a change, but cannot check it in, and another person (or role) can review and check in the change but cannot check out or edit. In the simplest form, this approach will halve the number of instances of repository entities. More complicated schemas can further reduce the volume.

The maker-checker approach can also be coupled with special deployment strategies (e.g. generation of pre-compiled “.adb” file) to further reduce the repository sized by eliminating physical repositories in production and the later stages of testing.

Blaze provides all building blocks (API) necessary to support this more robust solution. The following describes the solution via a use case.

8.6.3 Sample “Maker—Checker” Use Case

Simplest (sample) Maker-Checker Use Case:

1. Maker creates a new “defect” or “ticket” to track repository changes related to some change request and sets its status (management property/value holder) to “in progress”.
An instance of a repository filter with ID representing change request, i.e. a collection of filters becomes a system of record for change management
2. Maker checks out repository entries, changes them and adds their paths to the filter representing this defect
Normal Blaze check-out functionality: Add new criteria to the filter for each entity that changes to accommodate this request
3. Maker checks in modified entries and changes filter/request status (value holder/management property) to “request approval”
Normal Blaze check-in functionality
4. Checker finds all filter instances in “request approval” status, applies them to review changed entities
Normal Blaze query and project filter functionality
5. In order to promote changes, checker sets the status of filter to “in testing” or “deployed” and executes project “publish with update” functionality with filter applied, which moves changes to the next environment. From that point this step could repeat to promote to the next environments and change status to something else.
Normal Blaze project publish functionality

8.6.4 Best Practices and Design Considerations

It is important to consider that the above-described scenario is not supported through OOTB features of Blaze Advisor. RMA customizations will be required. Some client requirements have even required FICO to produce minor release of the Blaze Advisor product. The cost of implementation and support was can be quite high. Careful analysis of the requirements and consideration of the cost/benefit should be documented prior to committing to such a design.



8.6.5 Issues and Design Considerations with Sample Use Case

The following are known technical issues with the use case described above. FICO Professional Services has experience resolving these issues for our customers and should be consulted to provide guidance on Chubb's specific requirements.

1. Initiation of change:
 - No OOTB top-level RMA links/buttons to create and modify change requests. RMA customizations required.
 - All filters reside in the same "Filters" tab in the RMA. RMA customizations or a filter template will be needed if an easy way to structure them is required.
 - RMA User requirements typically need an easy way to find all filters by status, contributor, changed file, etc. This must be custom developed.
2. Implementation of change:
 - No "add to filter..." context command at the entry level, i.e. need to go to filter definition and then manually enter entry path to add it to the filtering criteria. This must be custom developed.
 - Multiple check-out/check-in operations when several people work on the same defect and need to add their entries to the same filter; possibility to lock change requests and prevent other users from modifying it (could also be a plus).
3. Submission of change:
 - Requires another filter check-out/check-in to change filter/request status (in progress -> request approval)
4. Review of change:
 - Product specifications would indicate this be implemented using queries. However, filters are necessary for the publish step below. Using filters to find entities is more complicated than queries because entities may reside in different folders and projects.
 - Visual comparison is a feature of a query, not a filter. Once filtered entity list is produced, customizations are required to run a visual comparison query to see all difference from a previous version on one page.
5. Approve and publish change:
 - No OOTB "publish" or "deploy" command available from the RMA. Customizations to the RMA or a separate publish utility needs to be implemented.
 - There are no fine granular authorization capabilities for role-based field access control – who can modify it, who can see one value in a dropdown vs. another based on the user role. A custom provider that is closely tied to the custom authorization manager needs to be developed
 - "Publish with update" can add or replace entities in the next environment but cannot remove them. "Publish and replace" can do everything. But be careful using publish with replace; it will erase every entity that is not part of the filter.
 - Some applications have requirements to promote changes to a released copy in the same repository rather than publish to another one. This necessitates another version of filter entity to check in.

Bottom line: Blaze has all the building blocks to support the maker-checker scenario. But, this approach carries a lot of customization work. Therefore the maker-checker scenario is only indicated when the



application is large, foreseen to have a high degree of volatility in changes, and scalability is foreseen to be an issue. Otherwise, FICO recommends clients to keep it simple and stick with OOTB functionality.

8.7 FICO Recommendations for CPI Rule Management

This section provides a summary of the findings from the onsite sessions where FICO worked with Chubb CPI and CSI to understand Chubb's requirements and formulate suitable recommendations. It provides FICO's recommendation for an enterprise approach to Business Rule Lifecycle Management for the Chubb enterprise and then specific recommendations for CSI, CPI Print and CPI Underwriting.

8.7.1 Overview

This section contains the following subsections

1. Review of Onsite Findings
2. FICO Recommendations for Chubb Enterprise
3. FICO Recommendations for Underwriting Rule Management
4. FICO Recommendations for Print Rule Management
5. FICO Recommendations for Chubb Specialty Lines

8.7.2 Review of Onsite Findings

One way to summarize the findings of the FICO/Chubb onsite assessments would be to rank the overall maturity of the CSI and CPI teams with respect to their experience and maturity applying BRMS technology.

Chubb Personal Lines (CPI) is really just getting started. While they have done an admirable amount of research and are learning about BRMS at a very fast rate, they have no hands-on experience. Their knowledge is based upon what they have been able to compile from conversations with others who have direct experiences with BRMS. On a maturity scale from 0-to-5, the fact CPI has done an excellent job of climbing the learning curve about BRMS theory would give them a 1. But the lack of any practical experiences indicates a maturity level of 0. Overall, their maturity would at best be characterized as 0.5. This is not bad. CPI has only been thinking about BRMS for about a year. You do not just fall into a maturity model at level 5. You have to climb the curve and CPI is progressing satisfactorily.

Chubb Specialty Lines (CSI) BRMS capabilities are very mature compared to most FICO customers and the overall Insurance industry. CSI has been working with BRMS since roughly 2006, which gives them 5 years of experience. On a maturity scale from 0-to-5, CSI's knowledge of Blaze Advisor, the BRMS methodologies, best practices, and industry trends is very good, a solid 4. Their implementation experience, application of methodologies, use of the COE, development of repeatable processes, etc., is good, a solid 3. Overall, their maturity would be somewhere north of 3.5.

The figure below qualifies the 0-to-5 BRMS maturity scale used above.

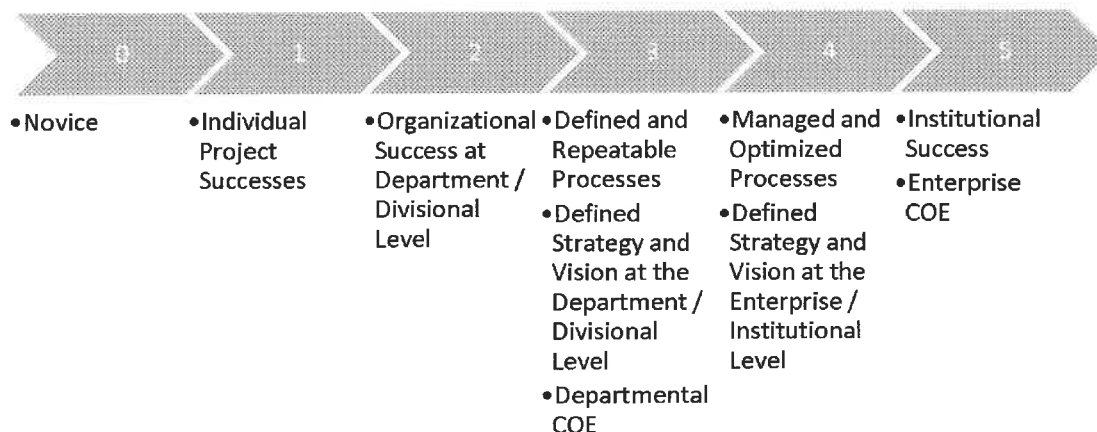


Figure 111—BRMS Maturity Scale

Interpreting the above assessment:

- CPI has an overall BRMS maturity score of 0.5, which places them well on the path toward being capable of delivering individual project successes.
- CSI has achieved defined and repeatable processes at the organizational level and are well on their way to having managed and optimized processes as well as a defined strategy and vision for BRMS within their enterprise.

8.7.2.1 Summary of Onsite with Chubb Personal Insurance (CPI)

CPI is just beginning their endeavor into the Business Rule Management Systems (BRMS). They are faced with several concurrent challenges:

1. Climbing the learning curve on using BRMS methodologies for managing decisioning within business software applications
2. Climbing the learning curve on Blaze Advisor's product features and capabilities
3. Accommodating some non-trivial challenges (i.e., unavailability of SME and complicated maintenance/release cycles) in their management of decisioning requirements

CSI is just beginning to digest some of the broader concepts of Business Rule Lifecycle Management and beginning to understand that it is a lifecycle vision and not simply version management. This is a natural progression and part of the normal process of adopting the BRMS methodology; there are several things about BRMS that appear a little bit strange at first and it takes some time for that to settle in.

8.7.2.1.1 Lifecycle Management Foundations during Discovery and Analysis

Both the Print and the Underwriting projects in CPI are modernizing legacy systems and will be endeavoring upon more of a rule mining effort than a decision requirements discovery and analysis effort. Neither of these projects is going to be ideal for establishing practices to lay the foundations of BRLM during the discovery and analysis phases. In this special case, FICO's recommendation would be to forego the establishment of any repeatable processes for these initial projects and begin to define process on the next suitable project.



8.7.2.1.2 Collaboration between Business and IT

Another of the non-trivial challenges that CPI faces is the practicality of involving its business in the management of the business rule lifecycle. This is something that is going to have to evolve over time through multiple iterations. It is not necessary to force collaborative analysis upon CPI and FICO sees no barriers or indications that an appropriate JAD relationship would not develop naturally after a couple BRMS implementations. This scenario is not unlike that experienced by the CSI teams who were early adopters of BRMS.

8.7.2.1.3 Environments, Staging, and Migration

CPI's environments, staging, and migration appear to be very consistent with that of CSI. There may be corporate standards in effect here. Nonetheless, the overview provided below for CSI is applicable to CPI. Please see below.

8.7.2.1.4 Simulation and Optimization

Simulation and Optimization are not in scope for either the Print or Underwriting projects.

8.7.2.1.5 Maintenance and Extension

The onsite assessment did not identify any maintainability or extensibility requirements for either Print or Underwriting that were unique to CSI or exceptionally challenging. CSI's challenges will be in the discovery and analysis of their decisioning requirements. However, once discovered, the implementation, maintenance, and extension should be relatively straight forward and in alignment with what FICO sees elsewhere in the Insurance industry.

That said, there are 2 key success factors or overarching design criteria that will be important for CPI.

1. While there is not expected to be a lot of volatility in decisioning requirements, and there is not expected to be a lot of complexity in the types of rule changes, there is expected to be a lot of (for lack of a better term) erratic behavior in the way changes are managed and transitioned into production. This erratic behavior includes fast-tracking, leapfrogging, tabling, suspending, sunseting, and many other non-sequential in the rule lifecycles.
2. The output of the new system needs to be precisely identical to the old system. That is not to imply that CPI intends to repave the cowpath. Simply, the new modern BRMS approach must produce the same results and the previous system.

FICO has experience and best practices for handling both of these design criteria. Success is not foreseen to be an issue, so long as considerations are made along the way.

8.7.2.2 Summary of Onsite with Chubb Specialty Insurance (CSI)

CSI is exceptionally experienced implementing BRMS with Blaze Advisor and has mature processes in place for managing the lifecycle of business rules. There has been reasonable progress toward a standardized approach across CSI. However, that progress has mostly precipitated from open communication between various teams. There is no managed process or program in place.

8.7.2.2.1 Lifecycle Management Foundations during Discovery and Analysis

CSI has made the connection that there is a direct relationship between the way decisioning requirements are discovered and harvested, and the success the project realizes with maintainability and extensibility. They are approaching the management of rule versions from a lifecycle perspective, not simply as a version management issue. They see that business rule lifecycle management begins with the early lifecycle of the business rules and is dependent upon the quality of the analysis work upfront.

Some major milestones accomplished by CSI over the past few years include:



- Moving toward a consistent approach for requirements capture across CSI
- Adapted FICO's DRAW Methodology for analyzing decisioning requirements
- Adapted FICO's Decision Analysis artifacts
- Adapted FICO's Decision Harvesting artifacts
- Moving toward consistent templates for artifacts across CSI

Current efforts are underway to grow maturity in several aspects of CSI's BRMS methodology

- Beginning to do volatility analysis up front, during rule harvesting. Currently focused more on rates of change rather than nature of change.
- Beginning to do analysis for reuse both within individual project and applications as well as across the CSI enterprise.

FICO recommends CSI begin to incorporate the following as their methodology matures

- Extend the analysis on how decisions are expected to change from the current analysis on the volatility of changes to include things like the nature of the changes, meaning how requirements will change, what aspects of the rule is volatile and what aspects are fixed.
- Include RMA requirements with decision harvesting
- Include unit test case/suite requirements with decision harvesting

8.7.2.2.2 *Collaboration between Business and IT*

CSI has also made the connection between the success of business rule lifecycle management (BRLM) and the hand-off between business and IT. CSI IT has developed collaborative relationship with their business and the two groups work closely during the hand-off. The business analysts also remain available to answer questions and provide elaboration throughout the implementation phases of the project.

The CSI IT/Business teams have adopted a JAD methodology and work together in back-and-forth iterations of discover, analysis, definition, design, implementation. They essentially run their BRMS implementation as mini-projects following an iterative/Agile methodology inside of the overall project's waterfall methodology. BRMS development is generally completed during the overarching project's design and development phases. Their goal is generally to have the rules in a near-final state by the end of the project's build phase, leaving only minor changes and tuning for Q/A.

This approach is very common throughout the BRMS industry. Many FICO customers use this approach to cohabitate the iterative methodologies required for BRMS development with the waterfall/cascade methodologies required for their corporate governance.

8.7.2.2.3 *Environments, Staging, and Migration*

CPI uses 5 environments in the SDLC.

1. Development—Includes prototyping, design, build, and unit testing
2. Integration Test—Includes integration and SOA testing
3. System Test—Includes systems testing and quality assurance (Q/A)
4. Model—Pre-production or staging environment for validation and regression testing of the environment. This environment is not consequential to BRLM.
5. Production—Production environment



BRMS Development is done in the Development environment, primarily in the Blaze IDE with some rules input via RMA for purposes of testing the RMA. Once the BRMS is stable and the most all the rules are input, the BRMS is promoted to the integration test environment.

Rule maintenance via the RMA can be done in the integration test, system test and production stages. The repository and RMA for these changes is said to be in the Production Environment. However, this is somewhat misleading.

There are no rule edits in the true production environment. The only BRMS artifact in the true production environment is a pre-compiled rule project (.adb file). This is also true for the integration test and system test environments. There is one repository supporting the .adb files deployed to all 3 of these environments and that repository resides in a separate pseudo-production environment reserved specifically for the BRMS. This BRMS environment is managed as a production asset but is not physically within the production environment.

CSI has expressed an interest in moving to a single repository architecture (combining the development and BRMS pseudo-production environments) and managing the business rule lifecycle through one overarching change management process. While this seems intuitively better, and there are some clear advantages to a single repository, CSI was not able to identify any specific business benefits or returns on investment. FICO has two concerns with this approach.

1. It eliminates a natural segregation and abstraction between the unstable/chaotic development environment and the stable/controlled production environment. Regardless of the amount of control, rigor, governance, etc., which is imposed upon a unified environment, accidents will happen. There are significant risks to this approach and CSI is not able to identify any substantial benefits.
2. This approach is predicated upon the expected behavior of a Blaze Advisor product feature that is not currently available. FICO PS recommends that CSI collaborate with FICO Product Management for Blaze Advisor to ensure that the ClearCase integration will support this approach. And then, re-analyze this approach after conducting a proof-of-concept.

8.7.2.2.4 Simulation and Optimization

CSI has made the connection between business rule lifecycle management and an ability to conduct simulations to validate and optimize the decisioning logic. They have built their own frameworks or "tools" for simulation that allows them to do business testing earlier than integration/SOA testing. This approach allows them to validate business logic during the iterative aspects of the project and release a more-fully tested decision service for integration/SOA testing.

These frameworks are, however, not generic tools. They are specific to the applications for which they were built. There is a simulation tool specifically for the Automated Renewal Processing project (ARP I) and a predictive modeling tool specifically for Specialty Claims. Some effort is underway to make a script-based tool more broadly applicable and reusable within CSI.

The obvious assessment from FICO's perspective is that CSI now has 3 disparate simulation frameworks that are not broadly applicable or reusable as is. FICO recommends a serious assessment on what it will take to grow one of these tools into an enterprise standard, and then maintain that tool for the enterprise. FICO's Decision Simulator might represent a steep initial investment with long-term returns in savings.

8.7.2.2.5 Maintenance and Extension

As mentioned earlier, CSI has a long history of using BRMS and has realized most of the advantages with respect to maintainability and extensibility of decision requirements offered by BRMS technologies. However, CSI has not yet made the transition to having the true business maintain and extend the business rules. At this point, such involvement of the business is not foreseen as a likely scenario.



CSI does not have a clear delineation between groups as primarily business and primarily technical. CSI's business groups often have technical staff on hand to implement and maintain small department-scale applications. CSI technical groups often have business analysts on hand to act as subject-matter experts and liaisons between the technical and business communities during the construction of large scale applications.

CSI is moving toward a model where the business analyst members of the IT community are taking on the responsibility for maintaining and extending business rules. IT business analysts are also responsible for review and approval of rules entered by technical staff during development. CSI sees this as a best practice for their organization. It is certainly aligned with what FICO sees happening at several other customers in the Insurance industry.

CSI sees three scenarios for IT business analysts to review rule requirements and rule changes with the business community, and then implement those rules or rule changes via the RMA:

1. Business community specifies rules and rule changes in English, BA inputs rules and rule changes using the RMA
2. Business community specifies rules and rule changes with spreadsheets, BA inputs rules or rule changes using the RMA
3. Business community implements rule changes directly using the RMA

8.7.3 FICO Recommendations for Chubb Enterprise

FICO's recommendation to Chubb for Business Rule Lifecycle Management at an enterprise level is to follow the pattern started by CSI and scale those processes to the enterprise level. CSI is doing most things quite well and the hiccups they face are relatively few road blocks.

1. FICO recommends Chubb initiate a program to proactively push information about updates for Blaze Advisor product features and capabilities as new versions of the product are released. This includes information about changes to best practices for proper BRMS structure and design. This will help ensure that all Blaze users are taking full advantages of product features and best practices that support management of the business rule lifecycle.
2. FICO recommends Chubb initiate a program to define corporate standards and practices for the following (in order to produce consistent designs that enhance the maintainability and extensibility of business rules)"
 - a. Corporate standard Blaze BRMS component architecture
 - b. Corporate standard Blaze Repository physical structure
 - c. Corporate standard Blaze Repository logical structure
 - d. Reusable user authentication and authorization (A&A) service
 - e. Recommended practices for lifecycle management for decision service
 - f. Recommended practices for lifecycle management for repository entities
 - g. Educate BRMS communities on the availability of product features and best practices for the use of Filters
 - h. Educate BRMS communities on the availability of product features and best practices for the use of Management Properties (Repository Entity Metadata)
 - i. Educate BRMS communities on the availability of product features and best practices for the use of Difference Query
 - j. Educate BRMS communities on the availability of product features and best practices for the use of Graphical Difference Tool



3. FICO recommends Chubb initiate a program to educate their BRMS community on best practices for rule versioning
 - a. Version Management System Considerations—the BRMS community needs to look at the lifecycle of rules, not just version management on the rule instances.
 - b. Versioning Rules, Rulesets, and Decision Metaphors—the BRMS community needs to take a more course-grain look at lifecycle management for decision metaphors. Instance-level versioning is not a common requirement and. Therefore, the preoccupation with row-level versioning may be indicative of a conceptual misunderstanding of decision metaphors.
 - c. Effective Dating—many of the rule lifecycle management challenges that Chubb faces can be handled with effective dating. The BRMS community needs to consider effective dating more often.
4. FICO recommends Chubb explore the FICO Lifecycle Management Framework and consider implementing it or a similar framework for
5. FICO recommends the Environment configuration depicted in Figure 112 below.
 - a. Chubb should continue to isolate development activities from core repository as presently practiced by CSI. The value of abstracting the chaos and volatility of development activities from the stable test and production rules more than offsets the additional management and administration effort around having two repositories.
 - b. As long as Chubb continues to follow a deployment strategy where pre-compiled Blaze Advisor projects (.adb files) are pushed to the runtime environments, they should continue to service the 3 testing environments and the production environment from one production BRMS.
6. FICO believes the Environment configuration depicted in Figure 112 below does not represent a significant departure from Chubb's current practices within CSI.



FICO Recommendations Whitepaper for Chubb Business Rule COE

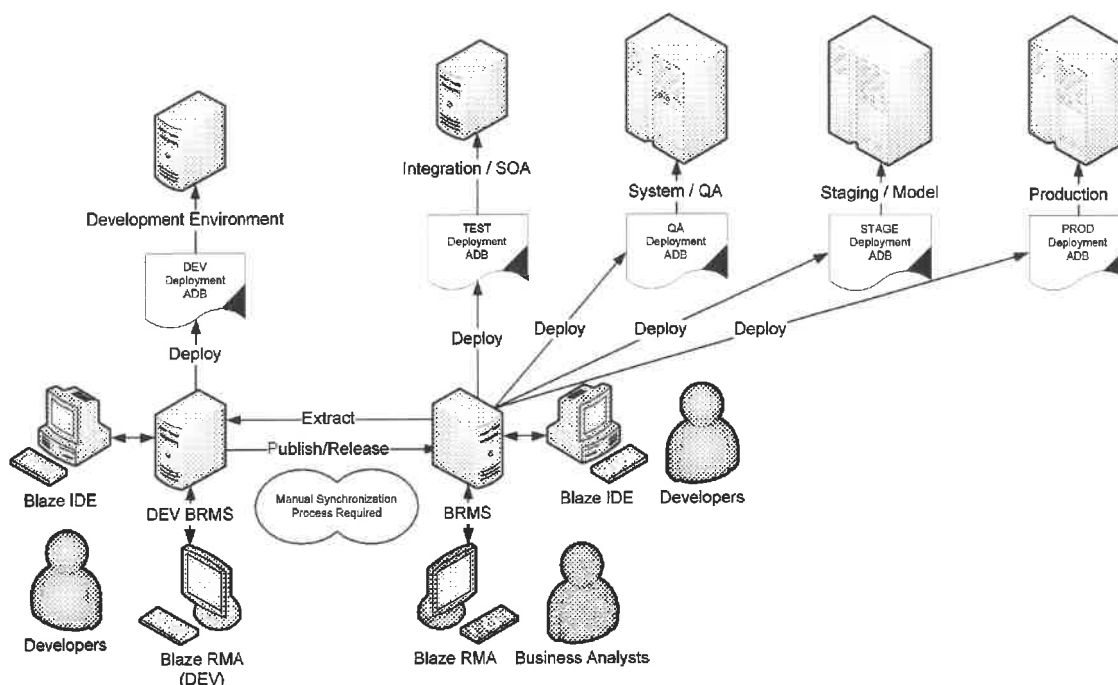


Figure 112—FICO Recommended Staging and Migration

8.7.4 FICO Recommendations for Underwriting Rule Management

This section is included in this document because it was specifically called out as a deliverable in the statement of work. FICO recommendations for Chubb Personal Insurance (CPI) in general, and the Underwriting project specifically, are discussed in Section 8.7.3 as part of FICO's recommendation for Chubb's broader enterprise.

8.7.5 FICO Recommendations for Print Rule Management

This section is included in this document because it was specifically called out as a deliverable in the statement of work. FICO recommendations for Chubb Personal Insurance (CPI) in general, and the Print Rule Management project specifically, are discussed in Section 8.7.3 as part of FICO's recommendation for Chubb's broader enterprise.

8.7.6 FICO Recommendations for Chubb Specialty Lines

This section is included in this document because it was specifically called out as a deliverable in the statement of work. FICO recommendations for Chubb Specialty Insurance (CSI) are discussed in Section 8.7.3 as part of FICO's recommendation for Chubb's broader enterprise.



FICO Recommendations Whitepaper for Chubb Business Rule COE

Appendix A

Attachments and Additional References

COBIT Management Strategies Whitepaper

March 31, 2011

FICO Confidential Page 258 of 259

Confidential

FED007129_0258



Appendix B

Notices/ Disclaimers

Confidentiality

FICO is providing confidential and proprietary information ("Confidential Information") in this whitepaper to Chubb Group of Insurance Companies ("Recipient") and Recipient may only use the Confidential Information to evaluate FICO's products and services. Recipient agrees to safeguard all Confidential Information disclosed by FICO with the same degree of care with which Recipient protects its own confidential information, which shall not be less than a reasonable standard of care. Recipient may disclose the Confidential Information on a "need to know" basis to its employees with whom Recipient has a confidentiality agreement in place to protect the Confidential Information. Recipient will not disclose such Confidential Information to any third party without the written consent of FICO. Upon request of FICO, Recipient shall destroy all tangible and electronic forms of the Confidential Information provided by FICO and Recipient shall not thereafter use or disclose the Confidential Information provided by FICO except as expressly permitted by FICO.

Agreement

This whitepaper is provided to assist Recipient in evaluating FICO's products and services. This whitepaper is not the basis of any agreement between the parties and does not form a contract between the parties. The actual terms and conditions of an agreement executed between Recipient and FICO will govern the license of products or purchase of services.

Validity

This whitepaper is valid for a period of 360 days from **3/31/2011**. Any extensions must be agreed in writing by FICO. FICO reserves the right to amend its prices at the time any extension is granted.

Trademarks

FICO and the FICO logo are registered trademarks of Fair Isaac Corporation in the United States and other countries. Other product names used herein may be trademarks or registered trademarks of Fair Isaac Corporation. Other product names and company names used herein may be trademarks or registered trademarks of their respective owners.